The Turing Degrees: Global and Local Structure [1]

Richard A. Shore ©

February 16, 2011

[1]THIS IS A ROUGH DRAFT. CORRECTIONS, COMMENTS AND SUGGGESTIONS ARE WELCOME. PLEASE SEND TO SHORE@MATH.CORNELL.EDU. PLEASE INCLUDE THE DATE OF THE DRAFT. THANKS.

Contents

Preface

Thank previous students and especially Mia Minnes who took notes in TeX for the entire course in 2007 along with the other students that term who all took turns preparing lecture notes.

Introduction

The introduction

0.1 Notation

$\mathbb{N}$ $a, b, c, d, e, i, j, k, l, m, n, r, s, t, u, v, w, x, y, z$

$\mathbb{N} \to \mathbb{N}$ f, g, h

$\mathbb{N} \to 2$ sets $A, B, C, U, V, W, X, Y, Z$

partial functions ϕ, φ, ψ...

Functionals Φ, Ψ, ... (continuous)

strings $\alpha, \beta, \gamma, \delta, \rho, \sigma, \tau$

String notation functional form if $\sigma = \langle x_1, \ldots x_n \rangle$ then $\sigma(i) = x_{i+1}$ (perhaps prefer $\langle x_0, \ldots x_{n-1} \rangle$); $\mathrm{dom}(\sigma) = n = |\sigma|$ length of σ; order by initial segments $\sigma \subseteq \tau$; restriction for $m \leq |\sigma|$, $\sigma \upharpoonright m \subseteq \sigma$ and $|\sigma \upharpoonright m| = m$. Apply to functions on all of $\mathbb{N}$ as well: $\sigma \subseteq f$; $f \upharpoonright m$.

Chapter 1

Beginnings

1.1 History, intuition, undecidability, formalization, ...

1.2 Formal definition, model of computation:

Turing machines (multitape with input, output and others)

Other notions: prim recursive + μ (search); register machines; equation calculus?..

1.3 Relative computability: Turing machines with oracles (on tape)

Recursive. (continuous) Functionals; oracles as inputs

1.4 Degrees, types of questions

algebraic, local...

second order, global: definability, automorphisms, theory

Chapter 2

Basics

2.1 Turing machines and relative computability

Consider Turing machines with oracles. There is master (universal) primitive recursive function,

$$\varphi(\sigma, e, x, s) = y$$

where the variables are σ a string (initial segment), e a number (index), x a number (input), s a number (steps, use); and the expression means that the Turing machine with index e and oracle restricted to σ given input x and run for s many steps converges and outputs y.

Conventions, Actions if ask question outside domain or no convergence in s steps: output $*$. Variables e, x, y range over natural numbers, $\mathbb{N}$.

Properties:

(i) Use: If $\sigma \subseteq \tau$ and $\varphi(\sigma, e, x, s) \downarrow = y$ then $\varphi(\tau, e, x, s) = y$

(ii) Permanence: If $s < t$ and $\varphi(\sigma, e, x, s) \downarrow = y$ then $\varphi(\sigma, e, x, t) \downarrow = y$

(iv) Domain of φ is computable, in other words there is a procedure to decide whether φ converges on any given tuple (σ, e, x, s). This procedure simply runs the machine with index e on input x and oracle σ. If the machine arrives at an output by step s, then answer yes (and otherwise, answer no).

Definition 2.1.1 *$\Phi_e^f(x) = y$ means that $\exists \sigma \subseteq f \exists s \big[\varphi(\sigma, e, x, s) \downarrow = y\big]$. So $\Phi_e^f(x)$ is a partial function (recursive in f). ??We define the* use of a computation *$\Phi_e^f(x) = y$ as the least n such that $\varphi(f \restriction n+1, e, x, s) = y$. We also say that $\sigma = f \restriction n$ is the* axiom *(about the oracle f) that gives this computation. Note that if f is changed at or below the use then this axiom no longer applies and no longer have the same computation giving the output y.*

Note that this definition can be recast in terms of sets, $\Phi_e^A(x) = y$

Definition 2.1.2 *$\Phi_{e,s}^X(x)$ means that we run Φ_e^X on x for s steps and report the output. So, $\varphi(\sigma, e, x, s) \equiv \Phi_{e,s}^{\sigma}(x)$.*

$\boxed{\texttt{useconv}}$ **Definition 2.1.3** *We adopt two conventions when the oracle is a finite string σ. First, we run the Turing machine for only $|\sigma|$ many steps so we write $\Phi_e^{\sigma}(x)$ for $\Phi_{e,|\sigma|}^{\sigma}(x)$. Second, we require that for $\Phi_e^{\sigma}(x)$ to converge we must have $x < |\sigma|$.(Roughly speaking we must read the input before giving an output.)*

Definition 2.1.4 *$f \leq_T g$ means $\exists e(\Phi_e^g = f)$.*

Intuitive definition that there is a Turing machine with oracle g that compute f. Argue that same. We use Φ_e to denote $\Phi_e^{\emptyset}$ the eth Turning machine with oracle the empty set (constant function 0). This is equivalent (explain) to the list of Turing machines without oracles and we will often simply identify these two versions.

Theorem 2.1.5 (s-m-n Theorem) *There is a one-one recursive function s_n^m such that*

$$\forall \bar{y}\left[\Phi_e^f(x_1, \ldots, x_m, y_1, \ldots, y_n) = \Phi_{s_n^m(e,\bar{x})}^f(\bar{y})\right].$$

In fact, can view m and n as variables as well.

Notion of uniformity?

Theorem 2.1.6 *Padding Lemma: $\forall e \exists^{\infty} i \forall f(\Phi_e^f = \Phi_i^f$*

Proof. Exercise: Informal argument and formal one using s-m-n Theorem. ■

Idea that s-m-n gives more: uniformity.

Theorem 2.1.7 *Enumeration Theorem: List of partial recursive (in f) functions: Φ_e^f.*

Theorem 2.1.8 (Recursion Theorem aka Fixed Point Theorem) *If f is a recursive function then there is an e such that for all g, $\Phi_e^g = \Phi_{f(e)}^g$.*

Kleene gave a one-line proof of the Recursion Theorem. But, it seems pretty magical. We will soon see a different proof..

Intuitively the recursion theorem implies that we can call a function h within the definition of f itself. This may seem counterintuitive or simply false. But think of the procedure that we envision defining h except that it has calls to h. Replace the calls to f by calls to Φ_e. this gives us a computation procedure whose index (as a Turing machine) is clearly recursive in e. Let f be the function computing the index of this procedure. By the recursion theorem f has a fixed point e. Now argue that Φ_e is a function (at least a partial function) as desired for h.

The Recursion theorem will also be used when we talk about approximation procedures.

Pairing functions: desiderata for $\langle x, y\rangle$.
$2^x 3^y$; $\frac{1}{2}(x^2 + 2xy + y^2 + 3x + y)$
$\langle x, y, z\rangle = \langle x, \langle y, z\rangle\rangle$ etc.
$\langle x_1, \ldots x_n\rangle = \langle n, \langle x_1, \langle x_2 \ldots\rangle\rangle\rangle$
Uniformity over length n.
$\prod p_i^{x_i+1} - 1$ for $\langle x_1, \ldots x_n\rangle$
"Define" uniformity by example.

2.2 Trees, Cantor and Baire space; topology; perfect sets

topology

trees of sequences from alphabet (formally identify with subset of $\mathbb{N}$)
binary trees, n-ary trees, finitely branching, f-branching
paths
Cantor space, Baire space
topology, open, closed, perfect sets
perfect trees
function trees

2.3 Partial orders and lattices

distributive
Boolean algebras
usl, lsl, susl, sls
universality issues
locally finite?
partial lattices??

2.4 Interpreting theories and structures

first and second order logic

Chapter 3

The Turing Degrees

Since $f \leq_T g$ is a transitive relation, we can define the equivalence classes. These are the Turing degrees, $\mathbf{f} = \{g | f \leq_T g \;\&\; g \leq_T f\}$. We then have the partial order $\leq$ on $\mathcal{D}$, the set of Turing degrees, induced by $\leq_T$.

Facts about the degrees:

- Has least element, $\mathbf{0}$, which is the degree containing all computable sets.
- There are elements of $\mathcal{D}$ other than $\mathbf{0}$. Two types of arguments. Counting and construction/definition.
- There are $2^{\aleph_0}$ sets (subsets of $\mathbb{N}$). By Cantor's theorem $2^{\aleph_0} > \aleph_0$. Moreover, since there are countably many Turing machines, each degree is countable as a set ($f \equiv_T f+c$ any constant function c) and has at most countably many predecessors. So, not only are there degrees other than $\mathbf{0}$, there are $2^{\aleph_0}$ many degrees. Specific ones given later (DNR functions and the Halting Problem to start.)
- There is no largest degree because for each degree $\mathbf{x}$, can find a DNR relative to $\mathbf{x}$ and it is not below it. Relativization
- $\mathcal{D}$ is an upper semi-lattice. Can define join $\vee$ on $\mathcal{D}$: On sets/functions it is defined by
  $$f \oplus g(2n) = f(n) \qquad\qquad f \oplus g(2n+1) = g(n);$$
  this is inherited by the degrees $\mathbf{f} \vee \mathbf{g}$ = degree of$(f \oplus g)$. Note that we can use the join operator to produce a degree strictly above each degree because can take join of a member of the degree with some DNR relative to it. Also by counting: take any g not recursive in f (only countably many) and consider $f \oplus g$.

Note that we denote degrees in boldface, $\mathbf{a}$ or $\mathbf{f}$ and sets or functions in lightface, A or f.

We summarize these facts as follows.

Theorem 3.0.1 *$\mathcal{D}$ is an uppersemilattice with 0 of size $2^{\aleph_0}$ with the countable predecessor property.*

We will see later (??) that every countable partial order and even uppersemilattice can be embedded in $\mathcal{D}$. This also holds for ones of size $\aleph_1$ (??). Indeed each is isomorphic to an initial segment (downward closed subset) of $\mathcal{D}$. For these results $\aleph_1$ is as far as we can go. There are models of ZFC in which $2^{\aleph_0} > \aleph_1$ with uppersemilattices (partial orders) of size $\aleph_2$ that cannot be embedded in (as initial segments of) $\mathcal{D}$. ??

Exercise 3.0.2 *There is a cofinal sequence of degrees if and only if CH (continuum hypothesis) holds in which case the sequence can be chosen to have order type $\aleph_1$.*

`graph` **Exercise 3.0.3** *Every degree contains a set (i.e. characteristic function). (Graph(f))*

Some more questions about $\mathcal{D}$: how tall is it? how wide is it? is it a lattice? ... Answers coming up.....

How do we "build" a nonrecursive function. We can "implement" the idea of the proof of Cantor's theorem that there are more functions on $\mathbb{N}$ than elements of $\mathbb{N}$, i.e. $2^{\aleph_0} > \aleph_0$. This idea is a really a procedure called a diagonal argue

We extract the crucial property in our setting in the following definition.

Definition 3.0.4 (DNR) *A function h is DNR (diagonally non-recursive) if $\forall n(h(n) \neq \Phi_n(n))$.*

`dnrnotrec` **Proposition 3.0.5** *If h is DNR then h is not recursive.*

Proof. By the diagonal argument... ■

We can now prove the recursion theorem.

Proof of Recursion theorem. Suppose not, i.e. $\forall e(\Phi_e \neq \Phi_{f(e)})$. [Such an f is called fix point free (FPF).] We try to build a recursive DNR h for the desired contradiction.

Since $\Phi_e(e) \neq \Phi_{f(e)}$ for every e, we only need to make $\Phi_{h(e)} = \Phi_{f(\Phi_e(e))}$ to get $h(e) \neq \Phi_e(e)$. "Obviously" (by the $s{-}m{-}n$ theorem), there is such a recursive h: given e find the index of the machine which first computes $\Phi_e(e)$ and if it converges then computes f of the value and begins mimicking the machine with that index. This gives the description a machine that computes $\Phi_{f(\Phi_e(e))}$ and so an index, $h(e)$ for it. Going from e to $h(e)$ is an intuitively computable procure. Formally, the s-m-n theorem shows that it is a recursive function. On the other hand, our assumption (that f is FPF) implies (as above) that h is DNR for the desired contradiction. ■

Now to recover the standard constructive version of the theorem that actually computes the fixed point (with the usual uniformity), note that the index k for h can be found recursively in that of f (again by the $s-m-n$ theorem). Now $\Phi_{h(e)} = \Phi_{\Phi_k(e)} = \Phi_{f(\Phi_e(e))}$ and so if we let $e = k$ then $h(k) = \Phi_k(k)$ is the desired fixed point: $\Phi_{h(k)} = \Phi_{\Phi_k(k)} = \Phi_{f(\Phi_k(k))}$.

Chapter 4

R.e. sets and the Turing jump

4.1 The Jump Operator

Definition 4.1.1 *On functions, define the jump as $f' = \{e : \Phi_e^f(e)\downarrow\}$.*

Proposition 4.1.2 *$f \leq_T g$ implies that $f' \leq_T g'$ and so the* jump operator *is well-defined on the degrees.*

Proof. Since $f \leq_T g$, there is i such that $f = \Phi_i^g$. So

$$e \in f' \Leftrightarrow \Phi_e^f(e)\downarrow \Leftrightarrow \Phi_e^{\Phi_i^g}(e)\downarrow$$

The s-m-n theorem gives a recursive one-one function k such that $\Phi^g_{k(e,i)} = \Phi_e^{\Phi_i^g}$. In particular,

$$e \in f' \Leftrightarrow \Phi^g_{k(e,i)}\downarrow \Leftrightarrow k(e,i) \in g'$$

and so $f' \leq_T g'$. Thus if $f \equiv_T g$ then $f' \equiv_T g'$ and the jump operator is well defined on $\mathcal{D}$. ■

Proposition 4.1.3 $f \leq_T f'$

Proof. We need to compute $f(n)$ using f'?

If f is a characteristic function A (i.e. a set)then we can decide whether $n \in A$ using A' by recursively finding an e such that $\Phi_e^A(e)\downarrow \Leftrightarrow n \in A$. Formally, we can appeal to the s-m-n theorem to get a recursive one-one function k such that for each n, $\Phi^A_{k(n)}(k(n))\downarrow \Leftrightarrow n \in A$. This gives the desired reduction.

We can now appeal to Exercise graph 3.0.3 for the theorem for all functions f. Or we can prove it directly. ■

Exercise 4.1.4 *Give a direct proof that $f \leq_T f'$ for all functions. Solution: Instead of finding index of machine which asks whether $n \in A$, find a machine such that*

$$\Phi^f_{k(n,m)}(z)\downarrow \Leftrightarrow f(n) = m.$$

Then successively ask $k(n,0) \in f'$?, $k(n,1) \in f'$?, etc. until find $k(n,m) \in f'$ in which case output $f(n) = m$. This procedure halts because of the assumption that f is a function, hence total.

Proposition 4.1.5 $f <_T f'$

Proof. It clearly suffices to find an $h \leq_T f'$ which is DNR^f, i.e.

$$\forall n\big(h(n) \neq \Phi_n^f(n)\big)$$

We compute h from f' as follows: Given n, ask if $\Phi_n^f(n)\downarrow$ (in other words, if $n \in f'$). If so, let $y = \Phi_n^f(n)$ and put $h(n) = y + 1$. If not, set $h(n) = 0$. ■

Conclusion: The jump is a strictly increasing, order preserving operator on the degrees.

The jump of the empty set is, of course, $\emptyset'$. By our identification of the Turing machines without oracles with those with oracle $\emptyset$, it is identified with the usual halting problem $K = \{e|\Phi_e(e)\downarrow\}$, the set of indices e of Turing machines which halt on input e. One often wants to consider alternate versions such as $K_0 = \{\langle x, y\rangle\,|\Phi_x(y)\downarrow\}$. We can consider this as an alternative version of K or of the jump in general because the produce sets of the same degree.

Exercise 4.1.6 *For every f, $f' \equiv_T \{\langle x, y\rangle\,|\Phi_x^f(y)\downarrow\}$.*

In fact, more is true as we shall see in ?? (1-1 equivalence) and ?? (recursive isomorphism).

4.2 Trees and König's Lemma

So we have two ways of "getting" nonrecursive sets - diagonalization and the halting problem. Have seen that the second computes an example of the first. What about the other way? Does every DNR function compute K? If not, what can we say about the needed complexity (if there is any)? We take a side trip to an example of reverse mathematics and a comparison of the "strength" of versions of a well known combinatorial principle: König's Lemma.

While there are many mathematical definitions of a tree (and we will see others later), for now we take a simple representation. Remembering that we are in the world of the natural numbers, it makes sense to sue (for now at least) the following definitions.

tree **Definition 4.2.1** *A* tree *T is a subset of $\mathbb{N}^{<\mathbb{N}}$, the set of finite strings of natural numbers, that is closed downward under the natural partial order $\sigma \subseteq \tau$: σ is an initial segment of τ. (Or in the functional notation $\sigma(n) = \tau(n)$ for every $n < |\sigma|$. (We use $|\sigma|$ to denote the length of the sequence σ or in the functional notation its domain with the ordering on $\mathbb{N}$ given by $\in$ on the usual set theoretic representations of the natural numbers.) The root* *of any of our trees is then the empty string (or set) $\emptyset$. A* binary tree *is a tree of binary sequences, i.e. a downward closed subset of $2^{<\mathbb{N}}$. A* finitely branching tree *is a tree T such that for every $\sigma \in T$ there are only finitely many $\tau \in T$ with $\tau \supset \sigma$ and $|\tau| = |\sigma|+1$. If T is a tree we say that a subset P of T is a* path on T *if P is infinite, linearly ordered and downward closed (with respect to $\subseteq$). The set of paths on T is denoted by $[T]$. The elements of a tree are often called* nodes *and ones with no successors in the tree,* leaves

Exercise 4.2.2 *If you know some general abstract definition of a (binary, finitely branching) tree, do all of ours satisfy the definition you know?*

Exercise 4.2.3 (Thought Problem) *Think about what a "converse" might mean. We are restricted to countable sets (trees) but can we think of any countable tree as ("isomorphic to") one of ours? In general, what does it mean to code mathematical structures in $\mathbb{N}$?*

$\boxed{\texttt{KL}}$ **Lemma 4.2.4 (König's Lemma)** *If T is an infinite, finitely branching tree then T has an infinite path.*

Proof. We "construct" a path P in T by recursion. At each step t we have a node σ_t in T of length t with infinitely many successors on T. We begin, of course, with the root $\emptyset = \sigma_0$ relying on the fact that T is infinite to satisfy our condition. If we have σ_t we consider its immediate successors in T. By assumption there are only finitely many and so one of them say $\sigma_t\hat{\ }x$ has itself infinitely many successors on T. We let z be the least such x and let $\sigma_{t+1} = \sigma_t\hat{\ }z$. It is clear that $P = \{\sigma_t | t \in \mathbb{N}\}$ is a path in T. $\blacksquare$

$\boxed{\texttt{WKL}}$ **Lemma 4.2.5 (Weak König's Lemma)** *If T is an infinite binary tree then T has an infinite path.*

Is this proof (of König's Lemma) constructive or effective? If not could there be one that is? Is it "easier" to prove Weak König's Lemma than the full one? Is it easier to construct a path in an infinite, binary tree than an arbitrary finitely branching one? What might these questions mean? Not every infinite tree has a path at all but what about arbitrary trees with paths? How hard is to construct one?

We begin with the first question. One way of making the question precise is to ask if every infinite finitely branching (or binary) recursive tree T has an infinite recursive path. Or more generally if every infinite finitely branching (or binary tree) T has an infinite path recursive in T. If so, we might also want there to be a uniformly effective procedure that produces such a path, i.e. an e such that $\Phi_e(T)$ is an infinite path in T for every finitely branching or perhaps every binary tree. The answer is no for all the versions and the proof is intimately connected to the notion of DNR functions. On the other hand, we claim that König's Lemma is more complicated than the weak version, i.e. it really is weaker. The analysis here is intimately connected to the jump operator.

Theorem 4.2.6 *There is an infinite recursive binary tree with no infinite recursive path.*

Proof. We want an infinite binary tree T such that for every $f \in [T]$, $f \in DNR$. If we did not have to make T recursive, we could simply take all the binary strings σ that satisfy the definition of a DNR function on their domains: $\{\sigma \in 2^{<\mathbb{N}} | (\forall n < |\sigma|)(\sigma(n) \neq \Phi_n(n))\}$. However, this set is not recursive (Exercise norectree ??). We can however, eventually recognize when a binary string σ fails to be in this set by seeing at some stage s that $\Phi_{n,s}(n) \downarrow = \sigma(n)$ for some $n < |\sigma|$. The picture for building the desired (or any) recursive tree is that we are effectively going along deciding which strings are in T. Say at stage s of our recursive construction we must decide for every binary string of length s if it is in T or not. (This makes T recursive.) We eliminate unwanted paths when we recognize that some σ has failed our test for being DNR. More precisely if at stage s we see that $\Phi_{n,s}(n) \downarrow = \sigma(n)$ for some $n < |\sigma|$ then no strings $\tau \supseteq \sigma$ are ever put into T at any stage $t \geq s$. Formally, $T = \{\sigma \in 2^{<\mathbb{N}} | (\forall n < |\sigma|)[\neg(\Phi_{n,|\sigma|}(n) \downarrow = \sigma(n)]\}$. By our basic facts about our master function φ, T is clearly recursive. Consider now any $f \in [T]$. If $f \notin DNR$ then there is some n and s such that $\Phi_{n,s}(n) \downarrow = f(n)$. By definition no $\sigma \subseteq f$ with $|\sigma| \geq n, s$ can be on T and so,of course, $f \notin [T]$ for the desired contradiction. ■

nonrectree **Exercise 4.2.7** *The tree* $S = \{\sigma \in 2^{<\mathbb{N}} | (\forall n < |\sigma|)(\sigma(n) \neq \Phi_n(n))\}$ *is not recursive.*

Theorem 4.2.8 *There is an infinite recursive finitely branching tree T such that every path in T computes $0'$.*

Proof. We want to code $0'$ into every infinite path f on a recursive tree T. Now T is a subset of $\mathbb{N}^{<\mathbb{N}}$ the set of all finite strings. In analogy with the previous construction, we might think of ourselves as beginning with the nonrecursive tree consisting of the single path f such that $f(n) = 0$ if $\Phi_n(n) \uparrow$ and $f(n) = s$ if s is the first stage t such that $\Phi_{n,t}(n) \downarrow$. We now want to turn this into a recursive, finitely branching tree T such that f is its only path. We follow the plan of keeping "bad" strings from extending to paths of the last construction and set $T = \{\sigma \in \mathbb{N}^{<\mathbb{N}} | (\forall n < |\sigma|)(\Phi_{n,|\sigma|}(n) \uparrow \;\Rightarrow\; \sigma(n) = 0 \;\&\; \Phi_{n,|\sigma|}(n) \downarrow \;\Rightarrow\; \sigma(n) = s$ where $\Phi_{n,s}(n) \downarrow$ but $\Phi_{n,s-1}(n) \uparrow)\}$. Now T is easily seen to be recursive from our basic facts about φ. Moreover by definition for each n there are at most two numbers r such that $\sigma(n) = r$ for any $\sigma \in T$ (0 and the first stage t such that $\Phi_{n,t}(n) \downarrow$). Thus T is finitely branching.

T is also infinite as, by induction, for every $\sigma \in T$ either $\sigma\hat{\ }0 \in T$ or $\sigma\hat{\ }s \in T$ for s the first stage t such that $\Phi_{n,t}(n) \downarrow$ (and perhaps both). We now claim that the f defined above is the only path on T. Suppose $g \in [T]$ and consider $g(n)$ for any n. If $\Phi_n(n) \uparrow$ then for every $\tau \in T$ with $|\tau| > n$, we must have $\tau(n) = 0$ by the definition of T. On the other hand, if $\Phi_n(n) \downarrow$ then let s be the first stage t such that $\Phi_{n,t}(n) \downarrow$. Again by the definition of T, if $\tau \in T$ and $|\tau| > n, s$ then $\tau(n) = s$. As $g \upharpoonright n + s \in T$, we must have $g(n) = s$ as required. ■

So solving the problem of finding a path in any infinite recursive finitely branching tree provides a calculation of $0'$. Note that one might say that T is 2-branching but it is not a binary tree under our current definitions. This is perhaps somewhat mysterious but an important distinction as well shall see. There are at most two immediate successors of each σ but we cannot recursively bound what they might be.

By relativization if we can find a path in every finitely branching tree, we can compute the jump operator. What about binary trees? It is by no means obvious, and indeed requires several ideas, to provide a proof but this is not the case for finding paths in infinite binary trees. How can we make this precise. We can capture the idea that it is "possible" to always be able to solve one problem (such as finding paths in infinite binary trees) without being able to solve another (finding paths in infinite finitely branching trees) by using the notion of a model. We understand 'being able" to include the idea that if we have some f then we have any $g \leq_T f$ and similarly if we have both f and g then we have $f \oplus g$. We make this precise by saying that there is a class $\mathcal{C}$ of functions closed under $\leq_T$ (and $\oplus$) such that such that for every $T \in \mathcal{C}$ that is (the characteristic function of) an infinite binary tree then there is an $h \in \mathcal{C}$ which is a path in T. So in $\mathcal{C}$ we can solve the first problem. On the other hand, there is an infinite finitely branching tree $T \in \mathcal{C}$ for which there is no path in $\mathcal{C}$. Thus we have a "model" in which every infinite binary tree has a path but not every infinite finitely branching tree has one. The proof of these assertions will come in ??.

In the other direction, as every binary tree is finitely branching, it is immediate that if every infinite finitely branching tree in $\mathcal{C}$ has a path then so does every infinite binary tree. Thus we will be able to conclude that solving the problem of finding paths for infinite finitely branching trees is strictly harder than the analogous problem for binary trees. This result is intimately related to a similar claim about how hard it is to prove Lemmas KL 4.2.4 and WKL 4.2.5 in the sense of what axioms are needed for the proof. This is the subject of reverse mathematics. We will return to such issues at a few points in this book. Survey or introductory articles include...????. The basic text is Simpson ??

Relations with finding a DNR function: $\mathrm{DNR}_2 \equiv$ FPF, DNR_k but DNR weaker? An example of reverse mathematics. Arbitrary trees much harder.

some exercises

Finding solutions for König's Lemma, even for recursive trees, requires more than $0'$. This is an example where closure under solving two problems is equivalent but one can't get by with a reduction that (effectively) transforms a problem of one type into one of the so that any solution of the second computes one of the first.

Medvedev and Muchnik degrees. ...For later after do INF$\equiv 0''$"

Exercise 4.2.9 *Show that every infinite, finitely branching tree T has a path recursive in T''. Build a recursive tree such that any path computes $0''$.*

Exercise 4.2.10 *Show that not every infinite tree has a path.*

Exercise 4.2.11 *Relation to compactness, topological and logical.*

4.3 Recursively enumerable sets

We began with the notion of what it means for a set or function to be computable (recursive). We now want to consider a weaker notion. The idea is that, for a set A, while we might not be able to decide if $n \in A$, we might nonetheless be able to list its elements. That is we might have a recursive function whose values are the elements of A (assuming $A \neq \emptyset$). For such sets we have a recursive way of enumerating its elements: $f(0), f(1), \ldots, f(n), \ldots$. So if $x \in A$ we eventually find out by enumerating that fact when we get to $f(n) = x$ for some n. If $x \notin A$ we may never discover that fact. (If we could, A would be recursive by $\aleph_1$

Definition 4.3.1 *The following equivalent conditions define the statement "$A \subseteq N$ is recursively enumerable (r.e.) in B":*

- A is the domain of a partial recursive in B function. Notation: $W_e^B = \operatorname{dom} \Phi_e^B$.
- A is the range of a partial recursive function.
- A is either the range of a total recursive in B function or is empty.
- A is either the range of a 1-1 recursive in B function or is finite.

Theorem 4.3.2 *A is recursive if and only if both A and $\bar{A} = \mathbb{N} - \mathbb{A}$ are r.e.*

Recall that $A' = \{e : \Phi_e^A(e) \downarrow\}$. So, A' is r.e. in A because it is the domain of the function that on input e runs the eth machine with oracle A with input e. We want to show that A' is the most complicated set r.e. in A in various precise ways.

We say that a set A is reducible to one B if there is some procedure that allows us to decide membership in A using membership in B. We have already met the most important and fundamental such reducibility that of Turing: $A \leq_T B$. We can compute the membership of A by asking questions about the membership of elements in B during the computation. It may adaptively determine which questions it asks based upon answers to previous questions. We now define some other notions of reduction which are stronger than that of Turing in the sense that they imply but are not, in general, implied by Turing reducibility.
Definition.

1. ***1-1 reducibility***$(\leq_1)$: $A \leq_1 B$ if there exists a one-one recursive function f such that $\forall x\ x \in A$ if and only if $f(x) \in B$.
2. ***m or many-one reducibility*** $(\leq_m)$: Same as one-one reducibility except f is an arbitrary (so possibly man-one) recursive function.
3. ***truth-table reducibility*** $(\leq_{tt})$: $A \leq_{tt} B$ if there exists a recursive function f such that $f(x)$ is a propositional formula σ in variables $p_1, ..., p_k$ such that for all x $x \in A$ if and only if B satisfies σ. EXPLAIN

4. ***weak truth-table reducibility*** ($\leq_{wtt}$): $A \leq_{wtt} B$ if there exists a recursive function f and a Turing machine Φ_e such that $\Phi_e^B = A$ and the use of computation in $\Phi_e^B(x)$ is at most $f(x)$ for all x. This is sometimes called ***bounded Turing reducibility*** ($\leq_{bT}$). EXPLAIN

Note that we have the following:

$$A \leq_1 B \to A \leq_m B \to A \leq_{TT} B \to A \leq_{wtt} B \to A \leq_{tt} B \to A \leq_T B.$$

Intuitively, we can think of the truth table reduction as giving a Boolean function which when given the answer to the oracle queries, will produce the final answer of the reduction. Note that all time bounded complexity classes are tt reductions

The first three reducibilities are total procedures in the sense that applied to any set they always produce a set as output. The final one is not. It is like a tt reduction but may be partial on some sets. In fact tt reducibility is characterized by its being total on all set inputs.

Theorem 4.3.3 (Nerode's Theorem) *$A \leq_{tt} B$ if and only if there is e such that $A = \Phi_e^B$ and Φ_e^X is a total (characteristic) function for every X.*

Proof. Since tt is total by definition, one direction is immediate. For the other direction, say Φ_e^X is total for all X and $\Phi_e^B = A$. What happens when we run $\Phi_e^X(n)$ for some unknown X? We can build a computation tree which branches (in two) whenever the program asks a question $m \in X$ with the branches corresponding to the possible answers 0 or 1 to this question. We terminate the tree when the Turing machine halts (when it gets the answers supplied along the route followed so far). Since the computation halts for every oracle X, all possible paths are are terminated so (using even Weak König's Lemma) the tree is finite. We can build a truth table that corresponds to this reduction (propositional variables encode branch points and return outputs at end of every path). This is effective and gives a truth table reduction from A to B. ■

Diagram

This theorem depends essentially on the fact that we restricted our attention to sets rather than all functions. One way of looking at this is that $2^{\mathbb{N}}$ is a compact space (Cantor space) but $\mathbb{N}^{\mathbb{N}}$ (Baire space) is not. (The paths through a binary tree form a closed (and so compact) set in Cantor space. Each node at which we terminate the tree determine an open set (all paths extending it). If they cover the space (no paths in the tree) then by compactness some finite subset of these open sets cover the space and so the tree is finite (all nodes are initial segments of one of the finitely many nodes determining the open sets that form the cover of the whole space. Another (equivalent) one related to our discussion in the last ?? section is that if we are dealing with binary trees (we branched to 0 or 1 depending on whether some number is in our set) then if every path terminates, the whole tree is finite. (The compactness of $2^{\mathbb{N}}$ is equivalent to WKL. (EXPLAIN). The theorem is not true?? for arbitrary functions in the oracle. They would allow for infinite branching in our computation tree and König's Lemma fails for arbitrary trees ($\mathbb{N}^{\mathbb{N}}$ is not compact).

We now want to show that $\emptyset'$ is the most complicated r.e. set. We could show that $A \leq_T \emptyset'$ for every r.e. set A but in view of these new reducibilities we have just defined we can hope for more.

Definition 4.3.4 *A set A is called an* r-complete set for class C *if A is in $\mathcal{C}$ and for every $B \in \mathcal{C}$, $B \leq_r A$.*

Proposition 4.3.5 *A' is 1-complete for the class of sets r.e. in A.*

Proof. We already know that A' is R.E. in A. So we only need to prove for all e, $W_e^A \leq_1 A'$. By definition, $x \in W_e^A$ iff $\Phi_e^A(x) \downarrow$. So, the s-m-n theorem gives a recursive one-one k such that $\Phi_e^A(x) = \Phi_{k(e,x)}^A(k(e,x))$. Hence, we have $x \in W_e^A$ iff $k(e,x) \in A'$. ■

Proposition 4.3.6 *If $B \leq_m A'$ then B is r.e. in A so for all A, B, $B \leq_{m(1)} A'$ if and only if B is r.e. in A.*

Proof. By definition of $\leq_m$, there is recursive f such that $x \in B$ implies $f(x) \in A'$ implies $\Phi_{f(x)}^A(f(x)) \downarrow$. So, we can use the s-m-n theorem to get that $x \in B$ iff $\Phi_i^A(x) \downarrow$ for some i, hence $B = W_i^A$. The rest of the assertion then follows from the previous Proposition. ■

We have seen that $A \leq_T B$ implies $A' \leq_T B'$. Now we present a similar result that links Turing-reduction with m(1)-reduction.

Proposition 4.3.7 ***Proposition 4.3.8*** *$A \leq_T B \Leftrightarrow A' \leq_m B'$ and $A \leq_T B \Leftrightarrow A' \leq_1 B'$.*

Proof. We first prove that $A \leq_T B \Rightarrow A' \leq_1 B'$. So, we want to determine whether $\Phi_x^A(x) \downarrow$ by asking a membership question in B'. We claim that $\Phi_x^A(x) \downarrow$ iff $\Phi_{f(x)}^B(f(x)) \downarrow$ for some recursive 1-1 function f. Why? because for each x, we can produce a machine with oracle B which ignores its input and computes $\Phi_x^A(x)$ by simulating the machine Φ_x^A and whenever it asks a question about A, compute A from B as given by assumption. This gives a recursive method for producing index $f(x)$, which can be made 1-1 by the Padding Lemma. (or use s-m-n)

Now we prove $A' \leq_m B' \Rightarrow A \leq_T B$. In contrast, it is *not* the case that $A' \leq_T B' \Rightarrow A \leq_T B$. (see ??)

Recall that $A \leq_1 A'$ (hence $A \leq_m A'$) because A is r.e. in A (it is the domain of procedure with oracle A which returns yes if $x \in A$ and loops forever otherwise). Likewise, $\bar{A} \leq_1 A'$, hence $\bar{A} \leq_m A'$, because $\bar{A}$ is r.e. in A.

By earlier converse, $A \leq_m A' \leq_m B'$ implies A is r.e. in B and $\bar{A} \leq_m A' \leq_m B'$ implies $\bar{A}$ is r.e. in B. Since A is recursive in B iff $A, \bar{A}$ are both r.e. in B (??), A is recursive in B. ■

limitlemma **Theorem 4.3.9 (Shoenfield Limit Lemma)** *$A \leq_T B' \Leftrightarrow \exists f \leq_T B$ such that $\forall x\big(A(x) = \lim_{s\to\infty} f(x,s)\big)$. Note that asserting that $\lim_{s\to\infty} f(x,s)$ exists means that $f(x,s)$ is eventually constant for fixed x.*

??Slogan: Effective in the jump just in case have eventually correct recursive approximation.

Proof. Say $A \leq_T B'$, in other words $A = \Phi_e^{B'}$ equating the set with the function means that the characteristic function of A is $\Phi_e^{B'}$. We want $f \leq_T B$ such that $\lim_{s\to\infty} f(x,s) = A(x)$. Certainly, $A(x) = \lim_{s\to\infty} \Phi_{e,s}^{B'}(x)$. This is recursive in B', but not in B. In order to make it recursive in B, we want to approximate the oracle B' recursively in B. Since $B' = \{e : \Phi_e^B(e)\downarrow\}$, $B'_s = \{e : \Phi_{e,s}^B(e)\downarrow\}$ is an approximation for B' recursive in B. In fact, $B' = \lim_{s\to\infty} B'_s$ because approximation changes at most once for each e.

We can approximate any W_e^B similarly by $W_{e,s}^B = \{n : \Phi_{e,s}^B(n)\downarrow\}$ and $\lim_{s\to\infty} W_{e,s}^B = W_e^B$. ??Extract notation??

So, define f by $A_s(x) = \Phi_{e,s}^{B'_s}(x) = f(x,s)$ (with the convention that if Φ hasn't answered by time s, return " no"). Then $f \leq_T B$. It remains to verify that $A(x) = \lim_{s\to\infty} f(x,s)$. Since $A = \Phi_e^{B'}(x)$, there is s such that $A(x) = \Phi_{e,s}^{B'}(x) = \Phi_{e,t}^{B'}(x)$ for all $t \geq s$. The computation of A only uses finite information about B, say $\sigma \subseteq B$. Moreover there is s_1 such that $B'_t(n) = B'(n)$ for all $n < |\sigma|$ (aka $B'_t \upharpoonright |\sigma| = B' \upharpoonright |\sigma|$) for all $t \geq s_1$, because of permanence and the properties of limits.

Conversely, suppose there is $f \leq_T B$ and $A = \lim_{s\to\infty} f(x,s)$. We want to show that $A \leq_T B'$. To find $A(x)$, we could start computing $f(x,0), f(x,1), f(x,2)\ldots$ and we know that eventually we get the right answer. But how do we know when to stop? By definition

$$\exists s \forall t > s\big(f(x,t) = f(x,s)\big)$$

and for this s, $A(x) = f(x,s)$. Define the following program recursive in B: $\Phi_{k(s)}^B(s)\downarrow$ iff $(\exists t > s)\big(f(x,s) \neq f(x,t)\big)$. Note that $\{s : \Phi_{k(s)}^B(s)\downarrow\} \leq_T B'$. We can apply the program iteratively: does f change after stage 0? If so, can find s_0 where it changes. Does it change after s_0? etc. This procedure halts because f is eventually constant (since it is a limit). ■

In applications of the Limit Lemma, without loss of generality we adopt the convention that we consider only functions f for which $\forall x(f(x,0) = 0)$.

Theorem 4.3.10 *A is r.e. in B iff there is $f \leq_T B$ such that for all x, $A(x) = \lim_{s\to\infty} f(x,s)$ and $f(x,s)$ changes at most once ($|\{s : f(x,s) \neq f(x,s+1)\}| \leq 1$).*

Proof. If there is such an f, let $\Phi_e^B(x)$ be the program which searches for an s such that $f(x,s) = 1$, and halts if it finds one. Then $A = \operatorname{dom}\Phi_e^B$ so A is r.e. in B. Conversely, if A is r.e. in B, then $A = \operatorname{dom}\Phi_e^B$ for some e. Let f be the function

$$f(x,s) = \begin{cases} 1 \text{ if } \Phi_{e,s}^B(x)\downarrow \\ 0 \text{ otherwise.} \end{cases}$$

Then $f \leq_T B$, $\lim_{s\to\infty} f(x,s) = A(x)$ and $|\{s : f(x,s) \neq f(x,s+1)\}| \leq 1$. ■

A is difference of sets r.e. in B if $\exists f \forall x f(x,s)$ changes at most twice. Then $A = C_0 - C_1$ both r.e. in B.

Continuing in this fashion, get the difference hierarchy (Putnam-Gold hierarchy).

Definition 4.3.11 *A is an n-r.e. set if there is a recursive function f such that for all x, $A(x) = \lim_{s\to\infty} f(x,s) = A(x)$ and $|\{s : f(x,s) \neq f(x,s+1)\}| \leq n$.*

We can connect this definition with difference of r.e. sets: A is n-r.e. iff

$$A = \begin{cases} (((W_{e_1} - W_{e_2}) \cup W_{e_3}) \cdots) - W_{e_n} & \text{if } n \text{ is even} \\ (((W_{e_1} - W_{e_2}) \cup W_{e_3}) \cdots) \cup W_{e_n} & \text{if } n \text{ is odd,} \end{cases}$$

where $W_{e_1}, \ldots, W_{e_n}$ are r.e. sets.

Definition 4.3.12 *A is ω-r.e. if there are recursive functions f, g such that $A(x) = \lim_{s\to\infty} f(x,s)$ and $f(x,s)$ changes at most $g(x)$ many times.*

Exercise 4.3.13 *Show that, for each $\alpha \leq \omega$, there are α-r.e. sets which are not β-r.e. for any $\beta < \alpha$. Hint: list all n-r.e.(for fixed n or the for all n uniformly) sets and diagonalize making only $n+1$ (finitely) many changes .*

Exercise 4.3.14 *Show X is ω-r.e. iff $X \leq_{tt} 0'$ iff $X \leq_{wtt} 0'$*

Note that in general, *tt* reducibility doesn't coincide with *wtt* reducibility. What do we know so far about the reducibilities?

Proposition 4.3.15 $\leq_T \neq \leq_m$, $\leq_T \neq \leq_1$

Proof. If $X \leq_m A$ and $X \leq_m \bar{A}$ and A is r.e. then X is recursive. Why? $X \leq_m A$ and A r.e. implies that X is re; $X \leq_m \bar{A} \Leftrightarrow \bar{X} \leq_m A$ so $\bar{X}$ r.e. as well. However, $0', \overline{0'} \leq_T 0'$, and $0'$ r.e. but not recursive. So $\leq_T \neq \leq_m$ and $\leq_T \neq \leq_1$. ■

Exercise 4.3.16 *Show that $1-1, m, tt, wtt, T$ are all distinct reducibilities. Hint: for wtt and T make list of the reductions (applied to some finite oracle). How hard is it to do this? Try for something recursive in $0'$ and then diagonalize. wtt but not tt is too hard. again list total tt-functions but now build both A and B in stages. In A put in only 0 except when might diagonalize. In B put in sequence of 1's of length the next e to diagonalize ending at a place where we will diagonalize in A and then at least one 0. (Fill in A with 0's until this point.) Then fill in B with 0's until force convergence so decide what to put into A. So for x to be in A must have $x \in B$ and $x+1 \notin B$ then check $B \upharpoonright x$ to see how many 1's in the list ending at x, say it is e, then compute how many 0's need to put into B to make $\Phi_e(x)\downarrow$ and find answer. $A(x)$ is the opposite.*

The Ershov hierarchy extends the difference hierarchy into the transfinite. If we exhaust the recursive ordinals produce precisely all the sets recursive in $0'$.

Recursively inseparable sets. Gödel's incompleteness theorem.

One-one equivalence same as recursive isomorphism. Explain, prove.

4.4 Arithmetic Hierarchy

Notion of language for first order arithmetic. Then for arithmetic. Tension between expressiveness and simplicity. For our purposes want language to be recursive (and so all typical syntactic properties are recursive) and each function and relation to be (uniformly) recursive (and so all quantifier free relations are recursive). On the other hand want to as much as possible to be expressible as "simply" as possible.

What at a minimum. Want say 0 then perhaps successor $s(x)$ and/or addition $x + y$. In what sense is addition definable from successor (by recursion; implicitly; second order)? We want to restrict definability to first order formulas. Note that multiplication, $x \cdot y$, is not definable from addition.

Presburger addition is decidable.

Peano arithmetic or even Robinson arithmetic is not. Gödel's incompleteness theorem. (forward reference to proof). Idea of representation of recursive functions so decidability would solve the Halting problem. So we need at least multiplication. Typically put in $<$ and 1 as well although they are definable from addition.

Exercise 4.4.1 *Define $<$ and 1 from $+, 0$ in arithmetic.*

May want to put in more to make all recursive functions easily definable.

Exercise 4.4.2 *With a recursive language (and interpretation as uniformly recursive functions and predicates) it is not possible to define all recursive functions by quantifier free formulas.*

So we need to go to formulas with at least one quantifier. We can make life simple by adding in one master recursive predicate for $\varphi(\sigma, e, x, s) = y$ (so capturing the partial function). It is then immediate that every recursive predicate and function is definable by an existential formula, i.e. one of the form $\exists x_1 \exists x_2 \ldots \exists x_n \theta$ where θ is quantifier free. Or we can cite the theorem of Matijasevich (Davis, Putnam and Robinson) solving Hilbert's 10th problem negatively by showing that every r.e. set W is the solution set for a polynomial (with many variables), i.e. there is a polynomial $p(x, \bar{y})$ such that $W = \{x | \exists \bar{y}(p(x, \bar{y}) = 0\}$.

The language of arithmetic has symbols $+, \times, <, 0, 1, \varphi(\sigma, e, x, s)$. The Σ_n, Π_n formulas of arithmetic are defined as follows:

- $\Sigma_0 = \Pi_0$ are quantifier free formulas
- Σ_{n+1}: $\exists \bar{x}(F(\bar{x}))$ for $F \in \Pi_n$
- Π_{n+1}: $\forall \bar{x}(F(\bar{x}))$ for $F \in \Sigma_n$

An intermediate route puts bounded quantifiers into the language ($\exists x < s, \forall x < s$) as well as a few select predicates or functions β for coding finite sequences (of variable length) and the corresponding projection functions. (Explanation and/or thought exercise.) If we do so, $\Sigma_0 = \Pi_0$ have only bounded quantifiers. Note that the predicates defined by such formulas remain recursive.

Prenex normal form. Collapse like quantifiers. Move bounded quantifiers past unbounded ones.

A relation is Σ_n or Π_n if it is defined by a Σ_n or Π_n formula. A relation is in Δ_n if it is defined by both a Σ_n and a Π_n formula. Note that the notion of Δ_n is semantic rather than syntactic.

Properties of $\Sigma_n, \Pi_n, \Delta_n$ Relations:

- If $A, B \in \Sigma_n$ then $A \cup B \in \Sigma_n$, $A \cap B \in \Sigma_n$, $\bar{A} \in \Pi_n$.
- If $A \in \Delta_n$ then $\bar{A} \in \Delta_n$.
- Σ_n is closed under projection. That is, if $A(x, y) \in \Sigma_n$ then $\{y : \exists x A(x, y)\} \in \Sigma_n$.
- Both Σ_n and Π_n are closed under bounded quantification. For $F \in \Sigma_n$,

  $$\exists x < sF \qquad \equiv \qquad \exists x\big(F(x) \wedge x < s\big),$$

  and

  $$\forall x < s\exists y_1 F \qquad \equiv \qquad \exists y\big(y \text{ is an } s\text{-tuple} \wedge \forall x < sF(x, \pi_x(y))\big).$$

  Note that this is sufficient because both checking tuple-hood and the projection functions are recursive so can use master function φ to represent them in our language.
- Uniformity.

We can relativize $\Pi_n^A, \Sigma_n^A, \Delta_n^A$ by adding a syntactic predicate $A(x)$ to the language and interpreting it in the semantics as the particular oracle set A.

Proposition 4.4.3 *When we add in extra unary predicates or function symbols, the truth of Σ_0 formulas (even with bounded quantifiers) depends only on the values of the predicates (functions) below some value which can be computed recursively in the formula.*

We now see that we can define the recursive predicates as simply as possible.

Proposition 4.4.4 $B \in \Sigma_1^A \Leftrightarrow B$ *is r.e. in* A.

Move proof here.

So the recursive predicates (sets) in A are precisely the ones that are Δ_1^A.

4.5 The Hierarchy Theorem

Theorem 4.5.1 (Post's Hierarchy Theorem) *1. $B \in \Sigma_{n+1}^A \Leftrightarrow B$ is RE in some Π_n^A set.*

2. $A^{(n)}$ is Σ_n^A m-complete for $n > 0$.

3. $B \in \Sigma_{n+1}^A \Leftrightarrow B$ is RE in $A^{(n)}$.

4. $B \in \Delta_{n+1}^A \Leftrightarrow B \leq_T A^{(n)}$.

Proof. We will need to use induction. Let's start with base case for (3), i.e. $B \in \Sigma_1^A \Leftrightarrow B$ is RE in A. Suppose $x \in B \Leftrightarrow \exists y F(x, y, A)$ where F has only bounded quantifiers. Note that a formula which only contains bounded quantifiers is recursive in A. Let $\Phi_e^A(x) \downarrow \Leftrightarrow \exists y F(x, y, A)$ be the function which checks each value of y in turn and return " yes" answer if it finds one. So, $B = W_e^A$ and is RE in A. Conversely, suppose B is RE in A. Then $B = W_e^A$. This means that $x \in B \Leftrightarrow \exists\sigma\exists y\exists s\big(\varphi(\sigma, e, x, s) \wedge \sigma \subseteq A\big)$. Note that $\sigma \subseteq A$ is a bounded quantifier formula so we have a Σ_1^A definition of B.

To prove (1): The base case is $B \in \Sigma_1^A \Leftrightarrow B$ is RE in some Π_0^A set. Above we showed that if $B \in \Sigma_1^A$ then B is RE in A, which is Π_0^A. Conversely, if B is RE in some other Π_0^A set, C, then since C is recursive in A, B is also RE in A so also use (3) to get $B \in \Sigma_1^A$.

For the induction, suppose $B \in \Sigma_{n+1}^A$. So $x \in B \Leftrightarrow \exists y F(x, y)$ where $F(x, y) \in \Pi_n^A$. In particular, B is $\Sigma_1^{F(x,y)}$ so is RE in $F(x, y)$ by the base case. Hence, B is RE in the Π_n^A set $F(x, y)$. Conversely, if B is RE in $W \in \Pi_n^A$, by the base case, B is Σ_1^W. So, $x \in B \Leftrightarrow \exists\sigma, y, s\big(\varphi(\sigma, e, x, s) = y \wedge \sigma \subseteq W\big)$ which is a Π_n^A definition.

To prove (2): We've previously shown that A' is the m-complete RE set. It remains to do the induction step. $A^{(n+1)} = \big(A^{(n)}\big)'$, which by the $n = 1$ case is the m-complete $\Sigma_1^{A^n}$ set. By induction, $A^n \in \Sigma_n^A$ so using (1) and that fact that being RE in X is the same as being RE in $\bar{X}$, we have that $A^{(n+1)} \in \Sigma_{n+1}^A$. For completeness, suppose $B \in \Sigma_{n+1}^A$. Then by (1), B is RE in some Π_n^A set C. So, B is RE in $\bar{C} \in \Sigma_n^A$. By the induction hypothesis, $A^{(n)}$ is Σ_n^A m-complete, so B is also RE in $A^{(n)}$. But X' is the 1-complete RE set, so $B \leq_m \big(A^{(n)}\big)' = A^{(n+1)}$.

To prove the induction step of (3): $B \in \Sigma_{n+1}^A$ if and only if B is RE in some Π_n^A set, C (by 1). This happens if and only if B is RE in $\bar{C} \in \Sigma_n^A$, which (by 2) happens if and only if B is RE in $A^{(n)}$.

For (4): $B \in \Delta_{n+1}^A \Leftrightarrow B \in \Sigma_{n+1}^A \cap \Pi_{n+1}^A \Leftrightarrow B$ is RE in $A^{(n)}$ and $\bar{B}$ is RE in $A^{(n)} \Leftrightarrow B \leq_T A^{(n)}$. ■

The hierarchy theorem tells us that one quantifier corresponds to one iteration of jump operator. For example, we have that if F is a predicate recursive in A, then $\exists x F \leq_T A'$ and $\exists x \forall y F \leq_T A''$.

Moreover, the hierarchy theorem also shows that the jump hierarchy is real: there are new sets at each levels. In particular, $A <_T A'$ implies that we have a strict hierarchy and $A^n \in \Sigma_n \setminus \Pi_n$. So we have $\Pi_n \neq \Sigma_n$ and

$$\Sigma_0 = \Pi_0 = \Delta_0 \subsetneq \Sigma_1 \subsetneq \Delta_1 \cdots$$

Diagram

4.5.1 Index sets

Define and samples

Exercise 4.5.2 *Prove that $\{e|W_e^A = \emptyset\}$ is 1-complete for Σ_1^A.*

Exercise 4.5.3 *Prove that $\{e|W_e$ is infinite$\}$ is 1-complete for Π_2^A.*

Tot **Exercise 4.5.4** *Prove that $\{e|\Phi_e$ is total$\}$ is 1-complete for Π_2^A.*

Exercise 4.5.5 *Prove that $\{e|W_e$ is cofinite$\}$ is 1-complete for Σ_3^A.*

Exercise 4.5.6 *Prove that $\{e|W_e$ is recursive$\}$ is 1-complete for Σ_3^A. Hint: movable marker argument to fix location for diagonalization if not cofinite.*

jumphier

4.6 Jump Hierarchies

We would like a sense of what it means for a set to be small, or near 0.

Definition 4.6.1 *X is* low *if and only if $X' = 0'$.*

This is as close as you can get to measuring smallness using the jump. It says that the jump of X is as small (low) as possible. In many ways, such low sets look like recursive sets.

If we consider sets below $0'$, it is easy to see what it means for its jump to be as big as possible.

Definition 4.6.2 *For $X < 0'$: X is* high *if and only if $X' = 0''$.*

Again, many constructions which can be done below $0'$ can be done (more carefully) below any high set. Can we extend these notions of smallness and largeness beyond the degrees first jump?

Definition 4.6.3 *$X \in L_2$ if and only if $X'' = 0''$; for $X < 0'$, $X \in H_2$ if and only if $X'' = 0'''$.*
$X \in L_n$ if and only if $X^{(n)} = 0^{(n)}$; for $X < 0'$, $X \in H_n$ if and only if $X^{(n)} = 0^{(n+1)}$.

Now we generalize to degrees not necessarily below $0'$ again trying to capture the idea that the jump of a set is a small (low) or as large (high) as possible.

Definition 4.6.4 *$X \in GL_1$ if and only if $X' = X \vee 0'$; $X \in GH_2$ if and only if $X' = (X \vee 0')'$.*
$X \in GL_n$ if and only if $X^{(n+1)} = (X \vee 0')^{(n)}$; $X \in GH_n$ if and only if $X^{(n)} = (X \vee 0')^{(n)}$.

Chapter 5

Embeddings into the Turing Degrees

5.1 Embedding Partial Orders in $\mathcal{D}$

So far the only degrees we know are $\mathbf{0}$ and the iterations of the jump beginning with $\mathbf{0}'$. Are there others? Is $\mathcal{D}$ a linear order? If not, how "wide" is it? How far away from being a linear order? Where do these other degrees lie with respect to the ones we already know? We begin answering these questions by considering what is perhaps the simplest question and showing that $\mathcal{D}$ is not a linear order.

Notation 5.1.1 *We write $A|_T B$, A is Turing incomparable with B, for $A \not\leq_T B$ and $B \not\leq_T A$.*

$\boxed{\texttt{KP}}$ **Theorem 5.1.2 (Kleene and Post)** $\exists A_0, A_1 (A_0|_T A_1)$.

How can we approach such a result. We will recast the desired properties of the sets we want to construct into a list of simpler ones R_e called requirements. Then we will choose an approximation procedure so that we can build a sequence of approximations $\alpha_{i,s}$ "converging" to A_i so that the information in an an approximation $\langle \alpha_{i,s} \rangle$ can be sufficient to guarantee that we satisfy one of the requirements in the sense that R_e will be true of any pair $A_i \supset \alpha_{i,s}$.
Proof. We will build A_0, A_1. The requirements necessary to guarantee the theorem are:

$$R_{\langle e,j \rangle} : \Phi_e^{A_j} \neq A_{1-j}$$

for all $e \in \mathbb{N}$, $j \in \{0, 1\}$. It is clear that if we the sets we construct satisfy each requirement then the sets satisfy the demands of the theorem. Our approximations in this case will be finite binary strings (so initial segments if characteristic functions) $\alpha_{j,s}$ such that $A_j = \cup_j \alpha_{j,s}$.

The construction will not be recursive because A_0, A_1 can't both be recursive and incomparable. But, the approximations won't change once defined at some x; in other words, $\alpha_{j,s} \subseteq \alpha_{j,s+1}$ so we get better and better approximations.

What actions will satisfy a requirement? Given $\alpha_{j,s}$ $(j = 0, 1)$, we want $\alpha_{j,s+1} \supseteq \alpha_{j,s}$ to guarantee that we satisfy $R_{\langle e,j\rangle}$. For definiteness, let $j = 0$. We want $\alpha_0 \supseteq \alpha_{0,s}$, $\alpha_1 \supseteq \alpha_{1,s}$ such that for any $A_0 \supseteq \alpha_0$, $A_1 \supseteq \alpha_1$, $\Phi_e^{A_0} \neq A_1$. In other words,

$$\exists x \neg \big(\Phi_e^{A_0}(x) \neq A_1(x)\big)$$

We can choose x as the first place x at which $\alpha_{1,s}$ is not defined (formally $x = \mathrm{dom}(\alpha_{1,s}) = |\alpha_{1,s}|$). Ask if $\exists \alpha_0 \supseteq \alpha_{0,s} \big(\Phi_e^{\alpha_0}(x) \downarrow\big)$. If so, we can choose " least" such α_0. Which ordering does the " least" refer to? We can make a master list of all convergent computations $\varphi(\sigma, e, x, t)$, i.e. $\{\langle \sigma, e, x, t\rangle : \varphi(\sigma, e, x, t) \downarrow\}$ and then " least" refers to least quadruple $\langle \alpha, e, x, s\rangle$ in this list.

Then, set $\alpha_{0,s+1} = \alpha_0$ and $\alpha_{1,s+1} = \alpha_{1,s}^{\wedge}(1 - \Phi_e^{\alpha_0}(x))$. By the use properties, if $A_0 \supseteq \alpha_0 = \alpha_{0,s+1}$ and $A_1 \supseteq \alpha_{1,s+1}$ then

$$\Phi_e^{A_0}(x) = \Phi_e^{\alpha_0}(x) \neq 1 - \Phi_e^{\alpha_0}(x) = A_1(x).$$

What if no such α_0 exists? We do nothing, i.e. we set $\alpha_{i,s+1} = \alpha_{i,s}$. This finishes the construction.

A general principle of our constrictions is do the best you can, and if you can't do anything useful, then do nothing and hope for the best (i.e. that what you can is enough). In this case, it *is* enough because if $A_0 \supseteq \alpha_{0,s}$ then $\Phi_e^A(x) \uparrow$. (If $\Phi_e^A(x) \downarrow$ for any $A \supseteq \alpha_{0,s}$ then the computation only requires finitely much information about A and so $\Phi_e^{\alpha}(x) \downarrow$ for some finite initial segment α of A. As $A_0 \supseteq \alpha_{0,s}$ we can certainly take this α to extend $\alpha_{0,s}$ as well if $\Phi_e^{A_0}(x) \downarrow$.) So $\Phi_e^{A_0}$ is not total and can certainly then not be the characteristic function of a set, i.e. $\Phi_e^{A_0} \neq A_1$.)

Thus we have actually verified that the construction satisfies all the requirements and so provides the desired sets. Consider $R_{\langle e,j\rangle}$. Look at the stage s at which we acted for this requirement. Either we did something (defined $\alpha_{i,s+1} \neq \alpha_{i,s}$) which guaranteed the requirement by guaranteeing that $\Phi_e^{A_j}(x) \downarrow \neq A_{1-j}(x)$ at some x; or we did nothing by setting $\alpha_{i,s+1} = \alpha_{i,s}$ but in that case we also guaranteed that the requirement is satisfied by making $\Phi_e^{A_j}(x) \uparrow$ for some x. ■

Questions:

1. How do we know that this construction keeps going...i.e. that there is no point from which we " do nothing". If that was the case, then both A_0, A_1 are finite — bad! Why doesn't this happen? Is it necessary to include another requirement to guarantee this: $Q_e : \alpha_{j,s} \geq e$ (these are easy to satisfy). Whenever we do act on a requirement, we make one of the α's longer and since infinitely often there is an index e which doesn't look at its oracle and outputs 0, at stage where we deal with requirement with index e, automatically extend the oracle approximation. Hence, both strings get extended infinitely often. This is a common phenomenon that constructions often do more than you expect that they do.

2. How complicated are A_0, A_1? We want a bound on their complexity, such as $A_0, A_1 \leq_T 0^{(n)}$ (this also gives definability properties). To determine what n is, let's look back at the construction. By recursion, we have $\alpha_{j,s}$. To calculate $\alpha_{j,s+1}$, we asked one question:
$$\exists \alpha_0 \supseteq \alpha_{0,s}\big(\Phi_e^{\alpha_0}(x)\downarrow\big),$$
a Σ_1 question so $0'$ can answer it and tell us which case we're in. The do nothing case is easy to do. For the other case, we have to enumerate the master list $\{\langle \sigma, e, x, t\rangle : \varphi(\sigma, e, x, t)\downarrow\}$, which we can do effectively. So, once $0'$ told us which case we're in, everything else is recursive. Hence, $A_0, A_1 \leq 0'$.

3. Where do A_0, A_1 lie in the jump hierarchy? Because of the symmetry of the construction, even though $A_0 \not\equiv_T A_1$, they should have some of the same properties. Are they low (or can we add something to the construction to make sure that they're low)?

   Recall: A_0 is low iff $A_0' \leq_T 0'$ iff $\{e : \Phi_e^{A_0}(e)\downarrow\} \leq_T 0'$.

   We can add a new requirement:
$$N_{e,j} : \text{make } \Phi_e^{A_j}(e)\downarrow \text{ if we can}$$
   Suppose that at stage s we are acting on $N_{e,0}$, have $\alpha_{j,s}$. Ask if
$$\exists \alpha_0 \supseteq \alpha_{0,s}\big(\Phi_e^{\alpha_0}(e)\downarrow\big).$$
   If the answer is yes, let $\alpha_{0,s+1}$ be the least such α_0 and let $\alpha_{1,s+1} = \alpha_{1,s}$. On the other hand, if the answer is no, then do nothing and put $\alpha_{j,s+1} = \alpha_{j,s}$ This is called forcing the jump.

   Claim 1: Construction is still recursive in $0'$. Why? Action for requirements $P_{e,j}$ are still the same. For $N_{e,j}$, $0'$ can answer the question $\exists \alpha_0 \supseteq \alpha_{0,s}\big(\Phi_e^{\alpha_0}(e)\downarrow\big)$.

   Claim 2: Can compute A_0' from $0'$. Why? Since the whole construction is recursive in $0'$, $0'$ can go along the construction until it gets to the stage s at which we act for $N_{e,0}$. Then, it sees what the construction does and can compute A_0' from this action.

   Claim 3: We can relativize the construction to any degree $\mathbf{x}$ to get incomparables A_j^X between X, X' such that $(A_j^X)' = X'$. By relativizing, we mean that at each part of the computation where we have oracle α_j, we instead have the oracle $X \oplus \alpha_j$. At the end, we build $X \oplus A_j$. The verification of the construction goes through as before.

Claim 4: It is easy to extend the construction to more than two incomparables. We can change the requirements to

$$P_{e,i,j} : \Phi_e^{A_j} \neq A_i \qquad i \neq j.$$

Thus, we can produce countably many low incomparables between 0 and $0'$.

Exercise 5.1.3 *Show that the sets A_i of the original construction are already low.*

We can strengthen the notion of lowness and prove a bit more:

Definition 5.1.4 *A is* superlow *if $A' \leq_{tt} 0'$.*

Exercise 5.1.5 *Prove that the sets constructed in Theorem* KP *5.1.2 are superlow.*

In general, given a countable partial order $\mathcal{P}$, can we embed it in $\mathcal{D}$ or in $\mathcal{D}(\leq 0')$ or in the low degrees? Let $\mathcal{P} = \{p_0, p_1, \ldots\}, \leq_{\mathcal{P}}$. Without loss of generality, we can assume that p_0 is the least element of $\mathcal{P}$. If $\mathcal{P}$ doesn't have a least element, add it and then embedding of this enlarged partial order gives embedding of suborder $\mathcal{P}$. We will build A_i such that $A_i \leq_T A_j$ if and only if $p_i \leq_{\mathcal{P}} p_j$. To do so, we build C_i and let $A_j = \oplus\{C_i : i \leq_{\mathcal{P}} j\}$. Does $i \leq_{\mathcal{P}} j$ imply that $A_i \leq_T A_j$? By transitivity,

$$\langle k, x\rangle \in A_i \;\Leftrightarrow\; x \in C_k \wedge k \leq_{\mathcal{P}} i \qquad \Rightarrow \qquad \langle k, x\rangle \in A_j \;\Leftrightarrow\; x \in C_k \wedge k \leq_{\mathcal{P}} j$$

so if $\leq_{\mathcal{P}}$ is recursive, $i \leq_{\mathcal{P}} j$ implies that $A_i \leq_T A_j$. We can use this fact to embed recursive partial orders in the low degrees by using the construction above to guarantee incomparability when needed and the recursiveness of $\mathcal{P}$ with this simple argument to guarantee comparability when needed. If a partial order is not recursive, it is at least recursive in some oracle so relativizing the proof for recursive partial orders gives embedding into $\mathcal{D}$. Perhaps this is the best we can do – it may not intuitively obvious that $\mathcal{D}(\leq 0')$ is a universal countable partial order.

Exercise 5.1.6 *The sets A_i constructed in the proof of Theorem* KP *5.1.2 are already low.*

$\mathbb{Q}$ as universal countable linear order. back and forth but we could construct one without knowing that $\mathbb{Q}$ has the desired property. We do it for partial orders.

Proposition 5.1.7 *There is a recursive universal countable partial order.*

Proof. Fraïssé. ■

As for linear orders there is a natural example

Proposition 5.1.8 *Every recursive partial order $\mathcal{P} = (P, \leq_{\mathcal{P}})$ with 0 can be embedded in $\mathcal{D}, \leq_T, 0$.*

Proof. Build sets C_i, define $A_i = \oplus\{C_j : p_j \leq_{\mathcal{P}} p_i\}$ so $p_k \leq_{\mathcal{P}} p_j$ implies $A_k \leq_T A_j$ since $\leq_{\mathcal{P}}$ is recursive.

Requirements: $R_{k,j,e} : p_k \leq_{\mathcal{P}} p_j$ implies $A_k \leq_T A_j$ i.e. $\forall e \Phi_e^{C_j} \neq C_k$.

Approximations: have finitely many finite binary strings $\gamma_{j,s}$. We will want $C_j = \cup\gamma_{j,s}$. Then approximate

$$A_{i,s} = \oplus\{\gamma_{j,s} : p_j \leq_{\mathcal{P}} p_i\}$$

i.e. $A_{i,s}$ is defined at $\langle j, x\rangle$ if $\gamma_{j,s}(x)$ is defined. Think of each $\gamma_{j,s}$ as partial function and $A_{i,s}$ is the sum of these partial functions. To ensure that A is a set (i.e. has a characteristic function), if $p_j \neq p_i$ make $A_j, i = 0$ (for totality)...CHECK INDICES.

Suppose we act for $R_{k,j,e}$ at stage $s = \langle k, j, e\rangle$. We have $A_{j,s}, A_{k,s}$ finite characteristic functions determined by the $\gamma_{i,s}$ so far defined. To guarantee $\Phi_e^{A_j} \neq A_k$, can we take $x = |\gamma_{k,s}|$ and ask if there is extension of the γ's such that $\Phi_e^{A_j}(x)\downarrow$? However, an extension of the γ's which guarantees convergence might also determine the value $A_k(x)$, so we might not be able to diagonalize!

To make x not interfere with A_k, want $x = \langle n, y\rangle$ such that $p_n \leq_{\mathcal{P}} p_j$. Also, to be able to define A_k, need $p_n \not\leq p_k$ (otherwise have empty column). And, need $\langle n, y\rangle \geq |\gamma_{k,s}|$. So we want $p_n \not\leq p_k$ and $p_n \not\leq_{\mathcal{P}} p_j$. By assumption, $p_k \not\leq_p p_j$, so choose $n = k$. Then, $x = \langle k, |\gamma_{k,s}|\rangle$.

Now, ask for least extension of γ's which makes $\Phi_e^{A_j}(x)\downarrow$ and only depends on γ_i for $p_i \leq_{\mathcal{P}} p_j$. If such an extension exists, put $A_k(x) = 1 - \Phi_e^{A_j}(x)$. If there is no such extension, do nothing. Then, go to stage $s+1$.

To verify that the construction satisfies all the requirements, for $R_{k,j,e}$ consider stage $s = \langle k, j, e\rangle$. Either we extended γ's or we didn't. If we extended, then there is x such that $\Phi_e^{A_j}(x)\downarrow\neq A_k(x)$. If we didn't then no such extension exists, and since A_j extends γ_s's, $\Phi_e^{A_j}(x)\uparrow$. ■

Corollary 5.1.9 *The one-quantifier theory of* $(\mathcal{D}, \leq_T)$ *is decidable.*

Proof. A one-quantifier sentence looks like

$$\varphi \equiv \exists x_1 \exists x_2 \cdots \exists x_n \big(x_i \leq x_j \wedge \cdots \wedge x_j \not\leq x_k \wedge \cdots \wedge x_n = x_n\big).$$

Note that if we can decide whether an existential sentence is true or false then we can flip the answers to decide if universal sentences are true and false. Given such a sentence, we can ask if there is a partial order that satisfies the sentence. If not, then $(\mathcal{D}, \leq_T)$ can't because it itself is a partial order. So suppose $(\mathcal{P}, \leq_{\mathcal{P}}) \models \mathcal{P}$. If we can embed $\mathcal{P}$ into $\mathcal{D}$ then we're done because embedding preserves atomic sentences. Not every partial ordering can be embedded into $\mathcal{D}$ (for example, huge ones can't). But if there is any partial order that satisfies φ then there is a finite partial order that satisfies it, because φ only mentions n elements. So, we can assume that $\mathcal{P}$ is finite, hence recursive. Then, the theorem above says that $\mathcal{P}$ embeds into $\mathcal{D}$. The last piece of the proof is to verify that we can answer the question of whether φ is satisfiable by a partial order. Well, we can enumerate all partial orders of size at most n and then check each one. And, if φ is satisfiable by a partial order then it is satisfiable by a member of the list. ■

Questions about proof of embedding theorem:

1. How complicated are the images of the partial order under the embedding? $A_i \leq_T 0'$ uniformly: to check $A_i(x)$ we ask if $x = \langle j, n\rangle \in A_i$. But, $\exists f \leq_T 0'\big(f(j,x) = \cup\gamma_{j,s}(x)\big)$. If $p_j \leq p_i$, can ask $f(j,n)$ what construction does. If $p_j \not\leq p_i$, ask $0'$ how construction goes. Hence, $A_i \leq 0'$ and indeed $\oplus A_i \leq 0'$.

2. Can we ensure that all A_i are low? We can add requirements

$$N_e : \Phi_e^{\oplus A_i}(e)\downarrow \text{ if we can.}$$

   To act on N_e still takes just a $0'$ question. Alternatively, instead of adding infinitely many requirements we can add a top element 1 to $\mathcal{P}$ and then construction gives $A_1 = \oplus C_j \leq 0'$ and then just make sure that A_1 is low.

Corollary 5.1.10 *The one-quantifier theory of $(\mathcal{D}(\leq_T 0'), \leq_T)$ is decidable.*

The method of finite approximations is used to build sets which are not necessarily though of in terms of Turing degrees.

Theorem 5.1.11 *There is a recursive partial order $\mathcal{P}$ such that every countable partial order $\mathcal{Q}$ can be embedded in $\mathcal{P}$*

Proof. We will build $\mathcal{P}$ by finite approximations, $\mathcal{P} = \cup\mathcal{P}_s$. At state s we have a finite partial order $\mathcal{P}_s$ and extend it to $\mathcal{P}_{s+1}$ such that for every subset of $\mathcal{P}_s$, every one element partial order extension is realized in $\mathcal{P}_{s+1}$. That is, for subset $M \subset P$, and a particular partial order relation on $M \cup \{z\}$ (z, a new element), add z to $\mathcal{P}$ and define its relation to the elements in $P \setminus M$ as that dictated by the axioms of partial orders. Thus we have the lemma that given any partial order and any finite subset and any extension by one element, there is a new partial order that realizes that extension. We can apply this finitely many times to take care of each finite subset and each possible one-element partial order extension. This construction is recursive so we have a recursive universal countable partial order.

Given $\mathcal{Q}$ a countable partial order, we use a forth argument to embed $\mathcal{Q}$ into $\mathcal{P}$. That is, if $\mathcal{Q} = \{q_0, q_1, \ldots\}$ we define the embedding f by induction. Start with $f(q_0) = p_0$ and then given $f \restriction n$, define $f(q_n)$ to be element of $\mathcal{P}$ realizing the extension of $f([n])$ that q_n does of $\{q_0, \ldots q_{n-1}\}$. ■

Note that if we could run the back direction as well so that $\mathcal{P}$ is embedded in $\mathcal{Q}$ so we have produced an ultrahomogeneous countable partial order. Other ultrahomogeneous structures are dense linear orders and atomless Boolean algebras.

Corollary 5.1.12 *Every countable partial order can be embedded in $\mathcal{D}(\leq_T 0')$.*

Proof. The universal partial order $\mathcal{P}$ above can be embedded in $\mathcal{D}(\leq_T 0')$ because $\mathcal{P}$ is recursive. By universality, any countable partial order can be embedded in it. ■

Definition 5.1.13 *$\{A_i : i \in \mathbb{N}\}$ is independent means that no A_i is computable from the join of finitely many of the other A_j.$\{A_i : i \in \mathbb{N}\}$ is very independent means that $A_i \not\leq_T \oplus_{j\neq i} A_j$ for all i.*

Very independent implies independent because $A_{i_1} \oplus \cdots \oplus A_{i_n} \leq_T \oplus_{j\neq i} A_j$ if no $i_k = i$ ($x \in A_i \Leftrightarrow \langle i, x\rangle \in \oplus_{j\neq i} A_j$) However, while independence is a degree theoretic notion, very independence is not. This is proved in the following exercises.

Exercise 5.1.14 *Find $\{A_i : i \in \mathbb{N}\}$ very independent. (Hint: either write down requirements and use finite approximations, or use partial order embedding).*

Exercise 5.1.15 *Find $\{A_i : i \in \mathbb{N}\}, \{B_i : i \in \mathbb{N}\}$ such that $\{A_i : i \in \mathbb{N}\}$ is very independent, $\{A_i : i \in \mathbb{N}\}$ is not, but $A_i \equiv_T B_i$.*

Definition 5.1.16 *An upper semi lattice is a partially ordered set $\mathcal{P}$ such that every pair of elements x, y in $\mathcal{P}$, has a least upper bound, $x \vee y$.*

Exercise 5.1.17 *Every usl $\mathcal{L}$ is locally countable, i.e. for any finite $F \subset L$ the subusl $\mathcal{F}$ of $\mathcal{L}$ generated by F (i.e. the smallest one containing F) is finite. Moreover, there is a uniform recursive bound on $|\mathcal{F}|$ that depends only on $|F|$.*

Exercise 5.1.18 *Given usls $Q \subset P$ and an usl extension $\hat{Q}$ of Q generated over Q by one new element (with $\hat{Q} \cap P = Q$), prove that there is an usl extension $\hat{P}$ of P containing $\hat{Q}$.*

Exercise 5.1.19 *Prove that there is a recursive usl $\mathcal{L}$ such that every countable usl can be embedded in it (as an usl).*

Exercise 5.1.20 *Every countable upper semi lattice $\mathcal{L}$ can be embedded in $\mathcal{D}$ and even in $\mathcal{D}(\leq \mathbf{0}')$ (preserving $\vee$ as well as $\leq$). Hint: Use a very independent set C_i. If $\mathcal{L} = \{l_i\}$ send l_i to $\oplus\{C_j | l_j \not\geq l_i\}$.*

Exercise 5.1.21 *Need definitions and hints: Alternatively the atomless Boolean algebra is countably universal for Boolean algebras, upper semilattices and partial orders. It also has a recursive representation as a lattice of recursive sets.*

We will see ?? that every countable lattice can be embedded in $\mathcal{D}$ but not by these methods in the sense that there is no countable lattice $\mathcal{L}$ which is countably universal, let alone a recursive one. Indeed local finiteness fails and there are $2^{\aleph_0}$ many lattices generated by four elements. (ref??)

What about uncountable partial orders, usls and lattices? Of course, they must have the countable predecessor property. It is known that all partial orders and even lattices of size $\aleph_1$ with the countable predecessor property can be embedded into $\mathcal{D}$. However, it is consistent that $2^{\aleph_0} = \aleph_2$ and there is an usl of size $\aleph_2$ with the countable predecessor property which cannot be embedded in $\mathcal{D}$. It is a long standing open question if every partial order of size $2^{\aleph_0}$ with the countable predecessor property can be embedded in $\mathcal{D}$. ref??

5.2 Extensions of embeddings

We now look at extensions of embedding results which give information about the 2quantifier theory of $(\mathcal{D}, \leq_T)$. Explain ....

Theorem 5.2.1 (Avoiding cones) *For every $A > 0$ there is B such that $A|_T B$.*

Proof. Given set A, we build B such that $A \not\leq_T B$, $B \not\leq_T A$. There are two kinds of requirements:

$$P_e : \Phi_e^A \neq B \qquad\qquad Q_e : \Phi_e^B \neq A.$$

The construction is by finite binary string approximations β_s for B. At the end, let $B = \cup_s \beta_s$.

Suppose at stage s we work to satisfy P_e. We have β_s and will construct β_{s+1} guaranteeing that B meets the requirement. Ask for value of $\Phi_e^A(|\beta_s|)$. If $\Phi_e^A(|\beta_s|)\uparrow$ then P_e is satisfied so do nothing. Otherwise, put $\beta_{s+1} = \beta_s\hat{\ }(1 - \Phi_e^A(|\beta_s|))$. So, $B(|\beta_s|) = \beta_{s+1}(|\beta_s|) \neq \Phi_e^A(|\beta_s|)$. Observe that we ask an A' question and then do a recursive procedure.

Likewise, suppose at stage s we work to satisfy Q_e. Ask if there is extension σ of β_s such that $\Phi_e^\sigma(|\beta_s|)\downarrow\neq A(|\beta_s|)$. If no such extension exists, do nothing. If there is an extension, let β_{s+1} be least such extension. Note that this is a Σ_1^A question followed by a recursive procedure, so this step is recursive in A'.

To verify that this works, observe that all P_e are clearly satisfied. Suppose we fail to satisfy Q_e. Then at stage s there was no extension $\sigma \supset \beta_s$ such that $\Phi_e^\sigma(|\beta_s|)\downarrow\neq A(|\beta_s|)$. But, if $\Phi_e^B(|\beta_s|)\uparrow$ then Q_e is satisfied. Therefore, $\Phi_e^B(|\beta_s|)\downarrow= A(|\beta_s|)$. But, this means that A is recursive. To compute $A(x)$, look for extension of β_s which makes $\Phi_e^\sigma(x)\downarrow$ and this must be correct value. This is a contradiction with our original hypothesis on A. Thus, Q_e is satisfied. ■

Exercise 5.2.2 *The B of the theorem can be made recursive in A' and indeed we can guarantee (or the construction already does) that $B' \equiv_T A'$.*

minpair **Theorem 5.2.3 (Minimal Pair)** *There are $A, B > 0$ such that $A \wedge B = 0$. In other words, for all C, if $C \leq_T A, B$ then $C = 0$.*

Proof. We build A, B by finite approximations α_s, β_s. There are three kinds of requirements:

$$P_e : \Phi_e \neq B \qquad Q_e : \Phi_e \neq A \qquad N_{e,i} : \Phi_e^A = \Phi_i^B = C \Rightarrow C \text{ is recursive.}$$

To satisfy P_e, Q_e (respectively): given α_s (β_s), ask if $\Phi_e(|\alpha_s|)\uparrow$ (or $\Phi_e(|\beta_s|)\uparrow$). If yes, then the requirement is already satisfied so put $\alpha_{s+1}(|\alpha_s|) = 0$ ($\beta_{s+1}(|\beta_s|) = 0$). Otherwise, put $\alpha_{s+1}(|\alpha_s|) = 1 - \Phi_e(|\alpha_s|)$ ($\beta_{s+1}(|\beta_s|) = 1 - \Phi_e(|\beta_s|)$).

Suppose at stage s we work on $N_{e,i}$. Ask if $(\exists \alpha \supseteq \alpha_s)$ $(\exists \beta \supseteq \beta_s)\exists x(\Phi_e^\alpha(x)\downarrow \neq \Phi_i^\beta(x)\downarrow)$. If such extensions exist, pick the first pair (α, β) which satisfy the condition and put $\alpha_{s+1} = \alpha$, $\beta_{s+1} = \beta$. If no such extensions exist, do nothing.

To verify that the construction works, first notice that all P_e, Q_e are satisfied so $A, B > 0$. For $N_{e,i}$, we may assume that $\Phi_e^A = \Phi_i^B = C$ as otherwise the requirement is automatically satisfied. We want to show that C is recursive; in particular, let's compute $C(x)$. Consider α_s, β_s for the stage s at which we work on $N_{e,i}$. To compute $C(x)$, find any finite extension $\alpha \supseteq \alpha_s$ such that $\Phi_e^\alpha(x)$. (There is one since $A \supseteq \alpha_s$ and $\Phi_e^A(x)\downarrow$.) We claim that $\Phi_e^\alpha(x) = C(x)$. If not, there is a $\beta \supseteq \beta_s$ with $\beta \subseteq B$ such that $\Phi_e^\beta(x) = \Phi_e^B(x) = C(x)$ and so we would have acted at s with α and β contrary to our assumption. ■

We will frequently use the idea see in this proof of searching for an extensions that give different outputs when used as oracles for a fixed Φ_e and if we find them doing some kind of diagonalization. If there are none, we generally argue that the Φ_e^A is recursive (or recursive in the relevant notion of extension as in Theorem exactpair 5.2.7). We extract the appropriate notion and provide some terminology.

Definition 5.2.4 *We say that two strings σ and τ e-split (or form an e-splitting) if $\exists x(\Phi_e^\sigma(x)\downarrow \neq \Phi_e^\tau(x)\downarrow$. We denote this relation by $\sigma|_e\tau$ and say that σ and τ e-split at x. Note that by our conventions in Definition useconv 2.1.3, $\Phi_e^\sigma(x) = \Phi_{e,|\sigma|}^\sigma(x)$ is a recursive relation as is $\exists x(\Phi_e^\sigma(x)\downarrow \neq \Phi_e^\tau(x)\downarrow)$, i.e. $\sigma|_e\tau$.*

Exercise 5.2.5 *We may make the A and B of the theorem low or note that as constructed they are already low. We can also relativize the result: $\forall C \exists A, B(A \wedge B \equiv C \ \& \ A' \equiv B' \equiv C')$.*

We want a similar notion to minimal pairs, but above any countable ideal of degrees rather than a single one.

Definition 5.2.6 *$\mathcal{C}$ is an ideal in the upper-semilattice $\mathcal{D}$ if $\mathcal{C}$ is closed under joins, and is closed downwards (i.e. if $Y \in \mathcal{C}$ and $X \leq_T Y$ then $X \in \mathcal{C}$).*

exactpair **Theorem 5.2.7 (Exact Pair)** *If $\mathcal{C}$ is any countable ideal in $\mathcal{D}$, there are A, B such that $\mathcal{C} = \{X : X \leq_T A, B\} = \{X : X \leq_T A\} \cap \{X : X \leq_T B\}$.*

An alternative statement of the theorem is the following:

Theorem 5.2.8 *If $C_1 \leq_T C_2 \leq_T \cdots$ is an ascending sequence, then there are A, B such that $\{X : X \leq_T A, B\} = \{X : \exists n(X \leq_T C_n)\}$.*

These two statements are equivalent because we can list all the elements D_j of a countable ideal and then consider the ascending sequence $C_i = \oplus_{j<i} D_j$. More ?? We will prove the second formulation of the theorem.

Proof. Given (C_n) ascending, we build A, B such that

- for all n, $C_n \leq_T A, B$ and
- $C \leq_T A, B$ implies that $C \leq_T C_n$ for some n.

Therefore, we need to satisfy the requirements

$$R_n : C_n \leq_T A, B \qquad\qquad N_{e,i} : \Phi_e^A = \Phi_i^B = C \Rightarrow \exists n(C \leq_T C_n).$$

We will build A, B by finite approximations α_s, β_s. But, instead of these being thought of as finite strings, they are matrices. In each matrix, finitely many columns are entirely determined, and there is finitely much additional information. Suppose at stage s we work for R_n. Choose the first column in each of α_s, β_s which has no specifications yet. Let α_{s+1} (β_{s+1}) be the result of putting C_n into that column of α_s (β_s) and leaving the rest of the approximation unchanged. This action is computable in C_n. Otherwise, suppose at stage s we work to satisfy $N_{e,i}$. Ask if $\exists x$ $(\exists \alpha \supseteq \alpha_s)$ $(\exists \beta \supseteq \beta_s)(\Phi_e^\alpha(x) \downarrow = \Phi_i^\beta(x) \downarrow)$. If such extensions exist, set $(\alpha_{s+1}, \beta_{s+1}$ to be the least such pair of extensions. If no such extensions exist, do nothing.

A, B meet the condition that for all n, $C_n \leq_T A, B$ because all R_n requirements are satisfied. Consider the stage s at which we deal with requirement $N_{e,i}$. We may assume that $\Phi_e^A = \Phi_i^B = C$ as otherwise the requirement is automatically satisfied. We want to prove $C \leq_T C_n$ for some n. Indeed let n be the largest m such that we have coded C_m into A and B by stage s. To compute $C(x)$, find any finite extension $\alpha \supseteq \alpha_s$ such that $\Phi_e^\alpha(x)$. (There is one since $A \supseteq \alpha_s$ and $\Phi_e^A(x) \downarrow$.) We claim that $\Phi_e^\alpha(x) = C(x)$. If not, there is a $\beta \supseteq \beta_s$ with $\beta \subseteq B$ such that $\Phi_e^\beta(x) = \Phi_e^B(x) = C(x)$ and so we would have acted at s with α and β contrary to our assumption. The crucial point now is that checking whether $\alpha \supseteq \alpha_s$ is recursive in C_n. ■

Exercise 5.2.9 *What is a bound on the complexity (degrees) of the A and B of the theorem in terms of the C_n? $(\oplus C_n)'$? How about a better bound? How low can we make this bound? If $C_n = 0^{(n)}$??*

extemb **Exercise 5.2.10 (Extensions of Embeddings)** *Given a finite usl $\mathcal{P}$ and a finite partial ordering $\mathcal{Q}$ extending $\mathcal{P}$ and with no $x \in Q - P$ below any $y \in P$ and an (usl) embedding $f : \mathcal{P} \to \mathcal{D}$ prove that there is an extension g of f embedding $\mathcal{Q}$ into $\mathcal{D}$. ??or prove??*

5.3 The range of the jump

5.3.1 The Friedberg Jump Inversion Theorem

This theorem describes the range of the jump operator on all the degrees.

Theorem 5.3.1 (Friedberg Jump Inversion Theorem)

$$\forall C \geq 0' \exists A (A' =_T C =_T A \vee 0').$$

The Friedberg Completeness theorem says that the only restriction on jump degrees is the obvious one: above $0'$. This was a small part of Friedberg's undergraduate thesis!
Proof. Let $C \geq_T 0'$. We build A by finite approximations α_s. The requirements are:

- $C \leq_T A'$ (coding C into A')
- $A' \leq_T C$ (keeping A' low)
- $A' \leq_T A \vee 0'$ (forcing the jump)

At stage s we have α_s. Ask if there is $\alpha \supset \alpha_s$ such that $\Phi_s^{\alpha}(s) \downarrow$. If so, we can choose " least" such extension α and let $\alpha_{s+1} = \alpha\hat{}C(s)$.

The construction is recursive in C (because $C \geq 0'$). So $\langle \alpha_s \rangle \leq_T C$. Moreover, $A' \leq_T C$ because $s \in A'$ iff $\Phi_s^{\alpha_{s+1}}(s) \downarrow$ (if $\Phi_s^A(s)$ converges, it is forced to by stage $s+1$). To check if $C \leq A' \leq A \vee 0'$ it suffices to check $C \leq_T A \vee 0'$. Moreover, it is enough to check that the construction is recursive in $A \vee 0'$. But, $0'$ can answer $\alpha \supset \alpha_s$ so then recursively look for least extension and then if we know that $A = \cup_s \alpha_s$, $\alpha_{s+1} = \alpha\hat{}C(s)$ is next element of A so $C(s) = \alpha_{s+1}(|\alpha_{s+1}|) = A(|\alpha_{s+1}|)$ and we can read it off A. Thus, $C \leq_T A \vee 0'$. ∎

Exercise 5.3.2 *Prove that all pairs of relations between A and B on one hand and A' and B' on the other not prohibited by the known facts that $A < A'$ and $A \leq_T B \Rightarrow A' \leq_T B'$ is possible.*

Exercise 5.3.3 *Jump inversion preserving partial order.*

Does not extend to preserving join. State noninversion theorem.

5.3.2 The Shoenfield Jump Inversion theorem

Is there a version of this theorem when we restrict the domain of the jump operator to degrees below $0'$? If $A \leq_T 0'$ then $0' \leq_T A' \leq_T 0''$ and A' is RE in $0'$. But maybe all sets above $0'$ are Turing equivalent to some RE set?

Exercise 5.3.4 *Prove that there is an $A \leq 0'$ such that $\forall e\,(A \not\equiv_T W_e)$ and so by relativization a C between $0'$ and $0''$ which is not r.e. in $0'$.*

Shjumpinv **Theorem 5.3.5 (Shoenfield Jump Inversion Theorem)** *For every $C \geq 0'$ which is r.e. in $0'$, there is an $A \leq_T 0'$ such that $A' \equiv_T C$.*

Proof. ?? ■

Prove this theorem. Hints: Work recursively in $0'$ and enumerate C. Use finite approximations to A. For coding C into A' use the Shoenfield limit lemma with the goal being that is $A^{[n]}$ is finite if $n \notin C$ and cofinite if $n \in C$. Use lowness type requirements to preserve computations of $\Phi_e^A(e)$.

trShjumpinv **Exercise 5.3.6** *Strengthen the Shoenfield jump inversion theorem by making $A <_T 0'$.*

Exercise 5.3.7 *Use the existence of nonrecursive low degrees, the previous exercise, relativizations and induction to prove the there are degrees in $\mathbf{L}_n$ and $\mathbf{H}_n$ for each $n \geq 1$.*

We can strengthen the notion of highness as we did that of lowness in Definition slow ??.

Definition 5.3.8 *$A \leq_T 0'$ is* superhigh *if $0'' \leq_{tt} A'$.*

Exercise 5.3.9 *If we take C in the proof of the Shoenfield jump inversion theorem to be $0''$ then the set A constructed in Exercise* StrShjumpinv *5.3.6 is superhigh.*

5.4 Trees and sets of size the continuum

There are several equivalent definitions of trees. One is that a tree is a connected undirected acyclic graph. Another is that a tree is a subset of $\mathbb{N}^{<\omega}$ closed under initial segments, ordered by $\subseteq$. (Note that binary trees are subsets of 2^ω; n-ary trees are subsets of n^ω.) A third definition is that a tree is a partially ordered sets with least element such that $\{x : x \leq a\}$ is linearly ordered for all a. Yet another definition uses the graph of function $F : S \to S$ with single fixed point (root) where each element has parent $F(x)$. Labelled trees are trees in any of these senses with an auxiliary function which labels each node.

A path through a tree is linearly ordered, closed downward (if there's an ordering). For the graph theoretic definition, a path through a tree is a path through the graph. Do we want to require that paths are maximal and/or infinite? Let's decide that paths are maximal, but need not be infinite. Denote by $[T]$ the set of paths through the tree T.

Theorem 5.4.1 *There is a set of pairwise incomparable degrees of size continuum, $2^{\aleph_0}$.*

Proof. We will build a tree with enough branching so that the number of paths through it is $2^{\aleph_0}$. In particular, since we wish to construct sets, we will build a binary tree $T \subseteq 2^{<\omega}$ such that if $A, B \in [T]$, $A \neq B$, then $A|_T B$. We want control over the structure of the tree: no dead ends, and perfect (every node has two incomparable extensions).

This guarantees that there are enough paths. Then, we identify a path P on T with $\cup P : \omega \to 2$, a characteristic function for some set.

The requirements on the tree are

$$R_e : \qquad \forall A, B \in [T] \forall e (\Phi_e^A \neq B).$$

To meet these requirements, we construct T by finite approximations. At stage s, we have a finite tree T_s. This finite tree has maximal elements $\sigma_1, \dots \sigma_n$ and any path through the finial tree T will have one of these as its initial segment. We consider what it means to meet a requirement for σ_i, σ_j. In other words, let $\sigma_{i,k}$ and $\sigma_{j,k}$ be extensions such that $\Phi_s^{\sigma_{i,k}}(x) \neq \sigma_{j,k}(x)$ (if convergent not equal, otherwise no extension converges). We can meet the requirement for s for each pair and finish in finite time (because there are finitely many maximal elements of a finite tree). This guarantees that if path A goes through σ_i and path B goes through σ_j then $\Phi_s^B \neq A$. Next, we split (add branching) by adding on $0, 1$ to each path. This gives T_{s+1}.

To verify the construction, suppose $A, B \in [T]$. There is $s, \sigma \in T_s, \tau \in T_s$ such that $\sigma \neq \tau$ and $\sigma \subseteq A, \tau \subseteq B$. Notice that by the Padding Lemma, there is $t > s, e$ such that $\Phi_t = \Phi_{e,s}$ so at stage t we ensured that $\Phi_e^A \neq B$. ■

Exercise 5.4.2 *There is a size continuum set of degrees which are pairwise minimal.*

Exercise 5.4.3 *There is an independent set of degrees of size continuum.*

This leads to many more embedding results. Using other ideas one can prove that any size $\aleph_1$ partial order with the countable predecessor property can be embedded into $\mathcal{D}$. it is an open question if every size continuum partial order with the countable predecessor property be embedded into $\mathcal{D}$.

Chapter 6

Forcing in Arithmetic and Recursion Theory

6.1 Notions of Forcing and Genericity

Forcing provides a common language for, and generalization of, the techniques we have developed so far. It captures the ideas of approximation to a desired object and how individual approximations guarantee (force) that the object we are building satisfies some requirement. Now approximations usually come with some sense of when one is better or gives more information than another. Of course, one approximation may have improvements which are incompatible , i.e. the set of approximations is partially ordered. The intuition is that $p \leq q$ means that p refines, extends or has more information than q. We are generally thinking that the conditions are approximations to some object $G : \mathbb{N} \to \mathbb{N}$ (typically a set) and that if $p \leq q$ then the approximation p gives more information than q and so the class of potential objects that have p as an approximation is smaller then the one for q. In addition, we have some notion of what at least at a basic level, the approximation p says about G. We formalize these ideas as follows:

forcing1 **Definition 6.1.1** *A* notion of forcing *is a partial order* $\mathcal{P}$ *with domain a set* P *and binary relation* $\leq_{\mathcal{P}}$. *For convenience, we assume that the partial order has a greatest element* $\mathbf{1}$. *(For further restrictions see Definition* forcing2 *6.1.16.)*

Example 6.1.2 *If the notion of forcing is* $(2^{<\omega}, \supseteq)$ *then* $\sigma \leq \tau \equiv \sigma \supseteq \tau$. *In many of our previous constructions we used such binary strings* σ *as approximations to a set* G *such that* $\sigma \subset G$. *So the longer the string, the fewer sets that "satisfy" it, i.e. have it as an approximation (initial segment). This example is often called* Cohen forcing.

Example 6.1.3 *In ?? we used finite binary trees with extension requiring that the extension add only strings that are extensions of leaves of the given tree. The object being approximated was a binary tree* T.

Example 6.1.4 *In ?? we used partial characteristic functions α defined on some initial segment of columns and some finitely many additional points. Again we were approximating a set $G \supset \alpha$.*

Example 6.1.5 *If the notion of forcing is the set of perfect (definition??) recursive binary trees under $\subseteq$ then $S \leq T \equiv S \subseteq T$. Think of a tree T as approximating the set $[T]$ of its paths so more information means fewer paths, i.e. more information about which path is being approximated. This notion of forcing is often called* Spector forcing *(or* perfect forcing *or* Sacks forcing *or other names for different variations).*

What is it or what class of objects is it that a condition p approximates? For Cohen forcing a condition (string) σ approximates the class of sets $\{G|G \supset \sigma\}$. So the collection of all approximations to a single set G is simply $\{\sigma|\sigma \subset G\}$, the class of all the initial segments of G. We want to isolate the salient features of this set of conditions or any set $\mathcal{G} \subseteq \mathcal{P}$ that might considered as an object its members are approximating. The general notion that we want for an arbitrary notion of forcing begins with that of a filter.

The idea is rather than comparing any two elements, compare them with the imaginary end point that we're approximating. That is, between current positions and end goal, there is an element.

Definition 6.1.6 *Two elements p, q are compatible if and only if $\exists r(r \leq p \wedge r \leq q)$. If p, q are incompatible we write $p \perp q$ (as opposed to incomparables which are written as $p \mid q$).*

Definition 6.1.7 *$\mathcal{F}$ is a filter on $\mathcal{P}$ if and only if $\mathcal{F}$ is upward closed its elements are pairwise compatible.*

Thus we are thinking of filters as connected with the object we are approximating.

Example 6.1.8 *Suppose we want to approximate a set $G \in 2^\omega$ and our notion of forcing is $(2^{<\omega}, \supseteq)$ (finite binary strings). Then the set $\{\sigma : \sigma \subset G\}$ is a filter. In particular, the union of this set (filter) is the characteristic function G. It will be common that the object we want is defined from a filter by some "simple" operation such as union. Note that for finite strings, being comparable is the same as being compatible.*

Example 6.1.9 *Suppose we want to approximate a set $G \in 2^\omega$ and our notion of forcing is some countable set of infinite binary trees (not necessarily perfect) such as the recursive ones. Then the set $\{T : G \in [T]\} = \{T : \forall\sigma \subset A(\sigma \in T)\}$ is a filter: Suppose two trees both have A as a path. Then the tree with just the path A is a common refinement. For upward closure, if A is a path on T and $T \subseteq S$ then A is also a path on S. In this case, the intersection of this filter is the characteristic function G.*

Suppose $\mathcal{F}$ is a filter on some notion of forcing $\mathcal{P}$. We can often associate some set or function with $\mathcal{F}$ in a canonical way. For example, for Cohen forcing we can naturally try $\cup\mathcal{F}$ For forcing with binary trees we might try $\cap\{[T]|T \in \mathcal{F}\}$. Does this always make sense even for Cohen or Spector forcing? For Cohen forcing it might be that $\cup\mathcal{F}$ is a finite string so itself a condition. For Spector forcing $\cap\{[T]|T \in \mathcal{F}\}$ could be a the set of paths through a binary tree with more than one branch which might not necessarily be recursive or perfect. We need to add conditions on our filter to make sure we get a total characteristic function, or a single set at the end. We might for example require for Cohen forcing that $\mathcal{F}$ contain strings of every (equivalently arbitrarily long) length, i.e. $(\forall n)(\exists \sigma \in \mathcal{F})(|\sigma| \leq n)$. For Spector forcing we could require that there are trees in $\mathcal{F}$ with arbitrarily long nodes σ before the first branching (i.e. σ has two immediate successors in the tree but no $\tau \subset \sigma$ does). We can represent meeting these requirements as getting into dense subsets of $\mathcal{P}$.

Definition 6.1.10 *$D \subseteq \mathcal{P}$ is* dense in $\mathcal{P}$ *if*

$$\forall p \in \mathcal{P} \exists q \in \mathcal{D}(q \leq_P p).$$

D is dense below r *if $\forall p \leq_P r \exists q \in \mathcal{D}(q \leq p)$.*

In general we want the conditions guaranteeing (forcing) each of our requirements to be dense.

Definition 6.1.11 *If $\mathcal{C}$ is a class of dense subsets of $\mathcal{P}$, we say that $\mathcal{G}$ is $\mathcal{C}$-generic if $\mathcal{G} \cap D \neq \emptyset$ for all $D \in \mathcal{C}$. We say that a sequence $\langle p_n \rangle$ of conditions is $\mathcal{C}$-generic if $\forall D \in \mathcal{C} \exists n(p_n \in D)$. All collections of dense sets considered are assumed to include the ones $\{p|\ |V(p)| \geq n\}$.*

seqfilter **Proposition 6.1.12** *If $\langle p_n \rangle$ is a $\mathcal{C}$-generic sequence then $\mathcal{G}$ =$\{p|\exists n(p_n \leq p\}$ is a $\mathcal{C}$-generic filter containing each p_n.*

Proof. $\mathcal{G}$ is $\mathcal{C}$-generic because it contains an element, p_n, of D_n for all n. It is upward closed because if $p \in \mathcal{G}$ then $p \geq p_e$ for some e so if $q > p \geq p_e$ and $q \geq p_e$ as well. Finally, it is pairwise compatible because given $p \geq p_{e_1}$, $q \geq p_{e_2}$ then $p, q \geq p_e$ where $e = \min\{e_1, e_2\}$. ■

Example 6.1.13 *For Cohen forcing let $D_n = \{\sigma : |\sigma| \geq n\}$ and consider $\mathcal{C} = \{D_n\}$. It is easy to see that this is a collection of dense sets. Then if filter $\mathcal{G}$ is $\mathcal{C}$-generic, it is guaranteed that $G = \cup\mathcal{G}$ is a set (i.e. defines a characteristic function $\omega \to 2$.*

If our collection of dense sets is countable then generic sequences and filters always exist.

Theorem 6.1.14 *If $\mathcal{C}$ is countable and $p \in \mathcal{P}$, then there is a $\mathcal{C}$-generic sequence $\langle p_n \rangle$ with $p_0 = p$ and so, by Proposition seqfilter 6.1.12, a $\mathcal{C}$-generic filter $\mathcal{G}$ containing p.*

Proof. Let $\mathcal{C} = \{D_n | n \in \mathbb{N}\}$. Then we define $\langle p_n \rangle$ by recursion beginning with $p_0 = p$. If we have p_n then we choose any $q \leq p_n$ in D_n as p_{n+1}. One exists by the density of D_n. It is clear that $\langle p_n \rangle$ is a $\mathcal{C}$ generic sequence and so $\mathcal{G} = \{p | \exists n(p_n \leq p\}$ is $\mathcal{C}$-generic filter containing p. ■

filterseq **Exercise 6.1.15** *If $\mathcal{C}$ is countable (as it always will be in our applications) and $\mathcal{G}$ is a $\mathcal{C}$-generic filter containing p, then there is a $\mathcal{C}$-generic sequence $\langle p_n \rangle$ with $p_0 = p$ such that $\mathcal{G} = \{p | \exists n(p_n \leq p\}$. (This is a converse to Proposition* seqfilter *6.1.12.)*

We thus always have a $\mathcal{C}$-generics for countable $\mathcal{C}$. We want to standardize and formalize the procedure of producing a generic object G (for us always a set or more generally a function from $\mathbb{N}$ into $\mathbb{N}$) from a generic filter or sequence. To do this we incorporate a function V associating approximations to G with conditions p.

forcing2 **Definition 6.1.16** *We always require that a notion of forcing have a function V into $\omega^{<\omega}$ which is recursive on P and continuous in the sense that if $p \leq_{\mathcal{P}} q$ then $V(p) \supseteq V(q)$. Moreover, we require that the sets $V_n = \{p| \ |V(q)| \geq n)\}$ are dense. (When we say that V* is recursive on P *we mean that it is the restriction of a partial recursive function to V which is defined on all of V.)??define elsewhere ?? We also require that any collection of dense sets that we consider for the construction of a generic filter or sequence include the V_n.*

As is our general practice, we will often care about how hard it is to compute a $\mathcal{C}$-generic. We must begin with the complexity of $\mathcal{P}$ and then consider how hard it is to compute the generic sequence $\langle p_e \rangle$ and finally the associated filter $\mathcal{G}$. We view the elements of $\mathcal{P}$ as being (coded by) natural numbers. For convenience we let the natural number **1** be the greatest element of P.

densityf **Definition 6.1.17** *A notion of forcing $\mathcal{P}$ is A-*recursive *(or* **a**-recursive*) if the set P and the relation $\leq_{\mathcal{P}}$ are recursive in A ($\in \mathbf{a}$). (As usual if $A = \emptyset$ ($\mathbf{a} = \mathbf{0}$) we omit it from the notation.) If $\mathcal{C} = \{\mathcal{C}_n\}$ is a collection of dense sets in $\mathcal{P}$ then f is a* density function for *$\mathcal{C}$ if $\forall p \in P \forall n \in \mathbb{N}(f(p,n) \in C_n)$.*

meetdense **Proposition 6.1.18** *If $\mathcal{P}$ is an A-recursive notion of forcing and $\mathcal{C} = \{\mathcal{C}_n\}$ is a uniformly A-recursive sequence of dense subsets of $\mathcal{P}$ and $p \in P$ then there is a $\mathcal{C}$-generic sequence $\langle p_n \rangle$ with $p_0 = p$ which is recursive in A. More generally, for an arbitrary notion of forcing $\mathcal{P}$, $p \in P$ and a class $\mathcal{C}$ of dense sets, if f is a density function for $\mathcal{C}$, then there is a $\mathcal{C}$-generic sequence $\langle p_n \rangle \leq_T f$ with $p_0 = p$. The generic G associated with these filters or sequences are also recursive in A or f, respectively.*

Proof. If $\mathcal{P}$ is an A-recursive notion of forcing and $\mathcal{C} = \{\mathcal{C}_n\}$ is a uniformly A-recursive sequence of dense subsets of $\mathcal{P}$, then we can define a density function $f \leq_T A$ by letting $f(p,n)$ be the least $q \leq_{\mathcal{P}} p$ with $q \in \mathcal{C}_n$. The desired generic sequence is now given by setting $p_0 = p$ and $p_{n+1} = f(n, p_n)$. As V is recursive and the dense sets V_n are by our conventions included among the $\mathcal{C}_n$, the associated generic set $G = \cup\{V(p_n)|n \in \mathbb{N}\}$ is recursive in f as required. ■

Note that the generic filter $\mathcal{G}$ defined from the generic sequence $\langle p_n\rangle$ in Proposition seqfilter 6.1.12 is Σ_1 in $\langle p_n\rangle$ but not necessarily recursive in it. While in the other direction (Exercise filterseq 6.1.15) the sequence is recursive in the filter and $\oplus C_n$.

[There are various connections between forcing, (generic) filters and topology. Order topology on $\mathcal{P}$...dense open sets , meager comeager, .generic..

In Cohen forcing the conditions correspond to (approximate) open sets in Cantor space 2^ω i.e. σ is an approximation to each set $G \supset \sigma$ and these form an open (even clopen) set in 2^ω. Then the intersection of all the clopen sets in a filter $\mathcal{F}$ is an open set. If the filter is mildly generic it is the single set G which is the union of the filter. In Spector forcing the intersection of the $[T]$ for T in some filter is a closed set. It is nonempty since the space is compact. If the filter is mildly generic the intersection is also a singleton.]

.

6.2 The Forcing Language and Deciding Classes of Sentences

An ad hoc approach to constructions is to look at the specific theorem we want to prove, decide what are the specific requirements we need to meet, and then build accordingly. For example, this is what we did to build $A|_T B$. Our approximations were $\mathcal{P} = \{\langle\alpha, \beta\rangle\}$. The requirements were $\Phi_e^A \neq B$ (and $\Phi_e^B \neq A$). Given α, β, we could find $\langle\hat{\alpha}, \hat{\beta}\rangle \leq \langle\alpha, \beta\rangle$ which would guarantee the requirement. In particular, if one exists, we chose $\langle\hat{\alpha}, \hat{\beta}\rangle \leq \langle\alpha, \beta\rangle$ such that $\exists x \Phi_e^{\hat{\alpha}}(x)\downarrow \neq \hat{\beta}(x)\downarrow$; if not, we took $\langle\alpha, \beta\rangle$. In the terminology of forcing, we had dense sets

$$D_e = \{\langle\alpha, \beta\rangle : \exists x \Phi_e^{\alpha}(x)\downarrow \neq \beta(x)\downarrow \text{ or } (\forall\langle\hat{\alpha}, \hat{\beta}\rangle \leq \langle\alpha, \beta\rangle)(\neg\exists x \Phi_e^{\hat{\alpha}}(x)\downarrow \neq \hat{\beta}(x)\downarrow)\}$$

Likewise, we define dense sets C_e, which guarantee $\Phi_e^B \neq A$. Then if $\mathcal{G}$ is $\{D_e, C_e\}$-generic, $G_0 \mid_T G_1$.

In this manner, each of the proofs we did earlier by constructions with requirements can be translated to dense sets and generics with the dense sets $\mathcal{D}_e$ determined by the conditions that guarantee (force) that we satisfy the eth requirement. (Exercises??) However, the benefit of the forcing technology comes in the form of the generality it allows. For example, we could try to tackle many of the constructions at once. We need to define the forcing relation ($\Vdash$) more generally, by induction on formulas φ that will somehow say that if $p \Vdash \varphi$ then $\varphi(G)$ holds for the set G determined by any sufficiently generic filter $\mathcal{G}$.

Thus we want a relation $\Vdash$ between elements $p \in \mathcal{P}$ and sentences $\phi(\mathtt{G})$ (where we use $\mathtt{G}$ as the formal symbol that is to be interpreted as our generic set G). This relation should approximate truth in the sense just described. We will use the language of arithmetic (in a set theoretic forcing, one would use the language of set theory) augmented with another parameter ($\mathtt{G}$) for the set we are building, and possibly other parameters (X, Y, A) for given sets. Recall that as in ?? we include bounded quantifiers and functions for coding and decoding sequences of numbers in our language of arithmetic. If desired we also include the recursive relation $\varphi(e, x, \sigma, s) = y$ which says that Turing machine e in input x with oracle σ when run for s many steps converges with output y. Note that the truth of $\varphi(G)$ for φ a Δ_0 formula depends on only finitely much of G and, indeed the amount of information needed is recursive in φ (and independent of G). (See ?? section on arithmetic.)

We use $\mathcal{G}$ for the generic filter, G for $\cup\{V(p)|p \in \mathcal{G}\}$, the set or function that we are building and $\mathtt{G}$ for the symbol in language that stands for that set or function. We will define the forcing relation $p \Vdash \varphi$ for $p \in \mathcal{P}$ and φ a sentence of our language by induction on the complexity of sentences.

Definition 6.2.1 *We define the relation p forces φ by induction.*

- If φ is Δ_0 formulas forcing is truth as far as $V(p)$ can determine it. By this we mean that $V(p)$ suffices to verify φ (in the sense of ??). Thus the forcing relation for Δ_0 sentences is a Δ_1 relation (in $\mathcal{P}$).

- For existential formulas:
$$p \Vdash \exists x \varphi \Leftrightarrow \exists n \left(p \Vdash \varphi(n)\right).$$

- For conjunctions and disjunctions:
$$p \Vdash \varphi \wedge \psi \Leftrightarrow p \Vdash \varphi \text{ and } \; p \Vdash \psi.$$
$$p \Vdash \varphi \vee \psi \Leftrightarrow p \Vdash \varphi \text{ or } p \Vdash \psi.$$

- For negated formulas:
$$p \Vdash \neg\varphi \Leftrightarrow \neg\exists q \leq p(q \Vdash \varphi).$$
or equivalently
$$p \Vdash \neg\varphi \Leftrightarrow \forall q \leq p(q \nVdash \varphi).$$

- For universal formulas: (remember that $\forall \equiv \neg\exists\neg$)
$$p \Vdash \forall x \varphi \Leftrightarrow \forall n \forall q \leq p \exists r \leq q \left(r \Vdash \varphi(n)\right).$$
or equivalently
$$p \Vdash \forall x \varphi \Leftrightarrow \forall n \forall q \leq p \left(q \nVdash \neg\varphi(n)\right).$$

deffor **Theorem 6.2.2** *If forcing for Δ_0 relations is recursive (in A) and the partial order is recursive (in A), then, for $n \geq 1$, forcing for Σ_n (Π_n) sentences φ (i.e. whether $p \Vdash \varphi$) is a Σ_n (Π_n) (in A) relation.*

Proof. We proceed by induction on n and for notational convenience ignore A. If $\varphi \in \Sigma_1$ then $p \Vdash \varphi$ is an existential quantifier applied to forcing of Δ_0 relations which by definition is Δ_1. Thus $p \Vdash \varphi$ is a Σ_1 relation for $\varphi \in \Sigma_1$. If $\varphi = \forall x\theta(x) \in \Pi_1$, consider the second version of the definition of $p \Vdash \varphi$. It says $\forall n \forall q \leq p\,(q \nVdash \neg\theta(n))$. As $\neg\theta$ is also Δ_0 $q \nVdash \neg\theta$ is Δ_1 and so $p \Vdash \varphi$ is Π_1. By induction and the definition for forcing an existential sentence, we see that for $\varphi \in \Sigma_{n+1}$ φ, $p \Vdash \varphi$ is a Σ_{n+1} relation. For $\forall x\theta(x) = \varphi \in \Pi_{n+1}$, $p \Vdash \varphi \Leftrightarrow \forall n \forall q \leq p \exists r \leq q\,(r \Vdash \theta(n))$. By induction $r \Vdash \theta(n)$ is Σ_n and so $p \Vdash \varphi$ is a Π_{n+1} relation. ■

ext **Exercise 6.2.3** *If $p \Vdash \varphi$ and $q \leq p$ then $q \Vdash \varphi$.*

Exercise 6.2.4 *??The order topology on a partial order $\mathcal{P}$ is defined by letting the sets of the form $\{q|q \leq p\}$ be the basic open sets. Show that as far as which formulas are forced by conditions in a $\mathcal{C}$-generic filter are concerned we may as well assume that all the dense sets in $\mathcal{C}$ are open as well.??*

We now want to tackle the question of how much genericity do we need to make forcing equal truth for generic filters/sets in the sense that if $p \Vdash \varphi$, $p \in \mathcal{G}$ and $\mathcal{G}$ is sufficiently generic then $\varphi(G)$ holds and, in the other direction, if $\varphi(G)$ holds then there is a $p \in \mathcal{G}$ such that $p \Vdash \varphi$.

Definition 6.2.5 *$\mathcal{G}$ is n-generic (for $n \geq 1$)iff for every Σ_n (in $\mathcal{P}$) subset S of $\mathcal{P}$,*

$$\exists p \in \mathcal{G}(p \in S \vee \forall q \leq p(q \notin S)).$$

We say that $\mathcal{G}$ is (ω-) generic if it is n-generic for all n.

The following equivalence is now immediate.

Proposition 6.2.6 *Let C_n be the class of sets $\{p : p \in S_e \vee \forall q \leq p(q \notin S_e)\}$ for all Σ_n (in $\mathcal{P}$) subsets S of $\mathcal{P}$. Then $\mathcal{G}$ is n-generic iff $\mathcal{G}$ is C_n-generic.*

Exercise 6.2.7 *If $D \subseteq \mathcal{P}$ is dense and Σ_n then D meets every n-generic $\mathcal{G}$. If D is dense below p and Σ_n then D meets every n-generic $\mathcal{G}$ containing p.*

To build an n-generic $\mathcal{G}$ we proceed as in the construction of a generic given a countable class of dense sets. We can calculate how hard it is to carry out this construction.

Proposition 6.2.8 *If forcing for Δ_0 sentences is Δ_1 (in $\mathcal{P}$), then, for each $n \geq 1$, there is an n-generic $G \leq_T 0^{(n)}$ ($\mathcal{P}^{(n)}$). There is also a generic $G \leq_T 0^{(\omega)}$ ($\mathcal{P}^{(\omega)}$).*

Proof. Exercise.?? ∎

Definition 6.2.9 *We say that a condition p decides a sentence φ if $p \Vdash \varphi$ or $p \Vdash \neg\varphi$.*

for=t **Theorem 6.2.10**

1. *If $\mathcal{G}$ is n-generic and $\varphi \in \Sigma_n$ then there is $p \in \mathcal{G}$ which decides φ. Moreover, if $p \Vdash \varphi$ then $\varphi(G)$ holds while if $p \Vdash \neg\varphi$ then $\neg\varphi(G)$ holds.*

2. *If $\varphi \in \Sigma_n$ (Π_n), $p \Vdash \varphi$ and $p \in \mathcal{G}$ which is n-generic then $\varphi(G)$ holds.*

Proof. We prove (1) by induction on $n \geq 1$. Consider $\varphi = \exists x\psi(x, \mathtt{G})$ with $\psi \in \Pi_{n-1}$. Now the set $S = \{p : p \Vdash \exists x\psi(x, \mathtt{G})\}$ is Σ_n by Theorem deffor 6.2.2. So by the definition of n-genericity, either there is $p \in \mathcal{G}$ in S, i.e. $p \Vdash \exists x\psi(x, \mathtt{G})$, or there is $p \in \mathcal{G}$ no extension of which is in S. If $p \in \mathcal{G}$ and $p \Vdash \exists x\psi(x, \mathtt{G})$, then (by definition) there is an n such that $p \Vdash \psi(n, \mathtt{G})$. Now by induction (or definition for $n = 1$), $\psi(n, G)$ holds and then so does $\exists x\psi(x, G)$ as required. On the other hand, suppose there is $p \in \mathcal{G}$ such that $(\forall q \leq p) q \not\Vdash \exists x\psi(x, \mathtt{G})$, i.e. $p \Vdash \neg\varphi$. In this case, we claim that that $\neg\exists x\psi(x, G)$. If not, there would be an n such that $\psi(n, G)$ and so by induction (or definition for $n = 1$), a $q \in \mathcal{G}$ such that $q \Vdash \psi(n, \mathtt{G})$. So, $q \Vdash \exists x\psi(x, \mathtt{G})$. But, since $p, q \in \mathcal{G}$ they are compatible and hence there is $r \in \mathcal{G}$ with $r \leq p, q$. This would contradict Exercise ext 6.2.3.

For (2) suppose $p \Vdash \varphi$ and $p \in \mathcal{G}$. If $\varphi = \exists x\psi(x, \mathtt{G})$ then $p \Vdash \psi(m, \mathtt{G})$ for some m and so by induction (or definition for $n = 1$) $\psi(m, G)$ holds. If $\varphi = \forall x\psi(x, \mathtt{G})$ but φ fails then for some m, $\psi(m, \mathtt{G})$ holds and so by (1) there is a $q \in \mathcal{G}$ such that $q \Vdash \psi(m, \mathtt{G})$. By the compatibility of $\mathcal{G}$ there would be an $r \leq_{\mathcal{P}} p, q$ in $\mathcal{G}$. This would again contradict Exercise ext 6.2.3. ∎

We will now look at degree theoretic properties of sets with various amounts of genericity. We begin with a connection between genericity and lowness.

Proposition 6.2.11 *If $\mathcal{G}$ is n-generic for Cohen forcing then $G^{(n)} = G \vee 0^{(n)}$.*

Proof. It is immediate that for any G, $G \vee 0^{(n)} \leq_T G^{(n)}$. Thus, it suffices to show that if $\mathcal{G}$ is n-generic then $G^{(n)} \leq_T G \vee 0^{(n)}$. The formula $\varphi(e, \mathtt{G})$ which says that $e \in \mathtt{G}^{(\mathtt{n})}$ is Σ_n. Therefore, by above theorem and the n-genericity of $\mathcal{G}$, either there is $p \in \mathcal{G}$ such that $p \Vdash \varphi(e, \mathtt{G})$ or there is $p \in \mathcal{G}$ such that $p \Vdash \neg\varphi(e, \mathtt{G})$. But forcing is Σ_n question and forcing negation is Π_n so to ask if $e \in G^{(n)}$ can search for $p \in \mathcal{G}$ such that $p \Vdash \varphi(e, \mathtt{G})$ or $p \Vdash \neg\varphi(e, \mathtt{G})$. This is a $G \vee 0^{(n)}$ question. By Theorem for=t 6.2.10, the one forced is the true fact about G. ∎

Exercise 6.2.12 *If $\mathcal{G}$ is 1-generic for any notion of forcing then $G' = G \vee 0'$.*

The next proposition gives almost all our previous incomparability and embeddability results in one fell swoop.

genindep **Proposition 6.2.13** *If $\mathcal{G}$ is Cohen 1-generic then the columns $G^{[i]} = \{\langle i, x\rangle\}$ form a very independent set, i.e. $\forall j(G^{[j]} \leq_T G_j)$ where we let $G_j = \oplus_{i \neq j} G^{[i]} = \{\langle i, x\rangle \in G | i \neq j\}$.*

Proof. For each e we want to show that $\Phi_e^{G_j} \neq G^{[j]}$. We consider the following set of conditions:

$$S_e = \{p : \exists x \left(\Phi_e^{p_j}(x) \downarrow \neq p^{[j]}(x)\right)\}.$$

Here we use the natural extension of our notation for columns of a set to finite binary strings: $p^{[j]} = \{\langle j, x\rangle \,|\, \langle j, x\rangle \in p\}$ and $p_j = \{\langle i, x\rangle \in p | i \neq j\}$. Since $S_e \in \Sigma_1$ and $\mathcal{G}$ is 1-generic, there is $p \in \mathcal{G} \cap S_e$ or there is $p \in \mathcal{G}$ no extension of which is in S_e. If $p \in \mathcal{G} \cap S_e$ then $p \subseteq G$ so the requirement is satisfied. Suppose that $p \subseteq G$ and $(\forall q \supseteq p) q \notin S_e$ then we claim that $\Phi_e^{G_j}$ is not total. If it were, let $\langle j, x\rangle$ be outside the domain of p. We must then have some $q \subset G$ with $q \leq p$ and $\Phi_e^{q_j}(x) \downarrow$. Now let $\hat{q}(\langle j, x\rangle) = 1 - q(\langle j, x\rangle)$ and $\hat{q}(z) = q(z)$ for $z \neq \langle j, x\rangle$. So $\hat{q}_j = q_j$ and so $\Phi_e^{q_j}(x) \downarrow = \Phi_e^{\hat{q}_j}(x) \downarrow$ but $\hat{q}(\langle j, x\rangle) \neq q(\langle j, x\rangle$ and so one of q and $\hat{q}$ (both of which extend p) is in S_e for the desired contradiction. ■

Relativization to X:

Expand our language to include a symbol interpreted as X. Define forcing in the same way as above and Σ_n genericity relative to X by using sets Σ_n in X. Theorem genthm ?? and Proposition genindep 6.2.13 then relativize to X. The Proposition now says that if G is 1-generic relative to X, then independence results hold even relative to X. That is, $\forall j(G^{[j]} \not\leq_T X \oplus G_j)$.

Exercise 6.2.14 *If G is Cohen 1-generic over X and $A, B \leq_T X$ then*

$$A \leq_T B \Leftrightarrow A \oplus G \leq_T B \oplus G.$$

Also, $G \,|_T\, X$ if $X > 0$.

Exercise 6.2.15 *We say that G_0 and G_1 are* mutually n-generic for $\mathcal{P}$ *if each G_i is n-generic over G_{1-i} for $\mathcal{P}$. Prove that if G is Cohen n-generic then the $G^{[i]}$ are very mutually Cohen n-generic in the sense that each $G^{[i]}$ is Cohen n-generic over $G^{[\hat{i}]} = \{\langle j, x\rangle \in G | j \neq i\}$. ??Note notation use everywhere??*

Exercise 6.2.16 *Translate the Exact Pair Theorem into the language of forcing. Hint: Given $\langle C_i\rangle$, define a notion of forcing $\mathcal{P}$ with conditions $\langle \alpha, \beta, n\rangle$ for $\alpha, \beta \in \mathbb{N}^{<\omega}$ and $n \in \mathbb{N}$. The ordering is given by $\langle \alpha', \beta', n'\rangle \leq \langle \alpha, \beta, n\rangle$ if $\alpha' \supseteq \alpha$, $\beta' \supseteq \beta$, $n' \geq n$ and, for $i < n$, if $\alpha'(\langle i, x\rangle) \downarrow$ but $\alpha(\langle i, x\rangle) \uparrow$ then $\alpha'(\langle i, x\rangle) = C_i(x)$ and similarly for β' and β.*

Exercise 6.2.17 *Construct a 1-tree T such that every $G \in [T]$ is Cohen 1-generic. Show that the Cohen 1-generic degrees generate $\mathcal{D}$. Hint: Consider any $F : \mathbb{N} \to \{0, 1, 2\}$ which is 2-generic for forcing over finite ternary ($\mathcal{P} = 3^{<\omega}$ with extension). Let d_n list the x such that $F(x) = 2$ in increasing order and, for $A \in 2^\omega$, let $F_A(x) = A(n)$ if $x = d_n$ for some n and $F_A(x) = F(x)$ otherwise. Show that F_A is Cohen 1-generic for every A. Let $F^{[j]}(x) = F(\langle j, x\rangle)$ and $F_A^{[j]}(x) = (F^{[j]})_A(x)$. Next show that for any j and A, $A \leq_T F_A^{[j]} \vee F_{\bar{A}}^{[j]}$. Finally show that for any $j \neq k$, $(F_A^{[j]} \vee F_{\bar{A}}^{[j]}) \wedge (F_A^{[k]} \vee F_{\bar{A}}^{[k]}) \equiv_T A$.*

Probably write out this proof as Theorem not exercise. $D = \{d_n | n \in \mathbb{N}\}$ is an *indifference set* ....??

latembsec

6.3 Embedding Lattices

latemb **Theorem 6.3.1 (Lattice Embedding Theorem)** *Every countable lattice $\mathcal{L}$ with least element 0 is embeddable in $\mathcal{D}$ preserving the lattice structure and 0.*

For later convenience, we actually want to prove an *a priori* stronger statement about partial lattices.

Definition 6.3.2 *A partial lattice $\mathcal{L}$ is a partial order $\leq_{\mathcal{L}}$ on its domain L together with partial functions $\wedge, \vee$ which satisfy usual definition when defined, i.e. if $x \wedge y = z$ then z is the greatest lower bound of x and y in $\leq_{\mathcal{L}}$; if $x \vee y = z$ then z is the least upper bound of x and y in $\leq_{\mathcal{L}}$. We say that $\mathcal{L}$ is recursive (in A) if L and $\leq_{\mathcal{L}}$ are recursive (in A) and $\vee$ and $\wedge$ are partial recursive (in A) functions on L with recursive (in A) domains.*

It may seem that there is no reason to use partial lattices but both effectiveness considerations and convenience come into play. It is certainly often more convenient to specify a partial lattice than to decide all the meets and joins.

Proposition 6.3.3 *If $\mathcal{L}$ is a partial lattice then there is a lattice $\hat{\mathcal{L}}$ and an embedding $f : \mathcal{L} \to \hat{\mathcal{L}}$ which preserves order and all meets and joins that are defined in $\dot{\mathcal{L}}$.*

Proof. Consider the lattice $\mathcal{I}$ of ideals of $\mathcal{L}$, i.e. subsets I of L closed downward and under join in $\mathcal{L}$ (when defined). The ordering on $\mathcal{I}$ is given by set inclusion. Meet is set intersection and the join of I_1 and I_2 is the smallest lattice containing both of them. The map that sends $x \in \mathcal{L}$ to $I_x = \{y \in L | y \leq_{\mathcal{L}} x\}$, the principle ideal generated by x, is the desired embedding into the sublattice $\hat{\mathcal{L}}$ of $\mathcal{I}$ generated by the principle ideals. ?? ■

??Not effective, i.e. if $\mathcal{L}$ is recursive $\hat{\mathcal{L}}$ isn't obviously so?? needn't be??Put in effective embeddings of p.o. and usl... all the way into Boolean algebras where do p.o. and usl embeddings??

To prove our theorem we need some lattice theory. In particular, we will use a type of lattice representations called lattice tables.

latrep **Definition 6.3.4** *A* lattice table *for the partial lattice $\mathcal{L}$ is a collection, Θ, of maps $\alpha : L \to \mathbb{N}$ such that for every $x, y \in L$ and $\alpha, \beta \in \Theta$*

1. *$\alpha(0) = 0$.*
2. *If $x \leq_{\mathcal{L}} y$ and $\alpha(y) = \beta(y)$ then $\alpha(x) = \beta(x)$.*
3. *If $x \not\leq_{\mathcal{L}} y$ then there are $\alpha, \beta \in \Theta$ such that $\alpha(y) = \beta(y)$ but $\alpha(x) \neq \beta(x)$.*

4. *If $x \vee y = z$, $\alpha(x) = \beta(x)$ and $\alpha(y) = \beta(y)$ then $\alpha(z) = \beta(z)$.*

5. *If $x \wedge y = z$ and $\alpha(z) = \beta(z)$ then there are $\gamma_1, \gamma_2, \gamma_3 \in \Theta$ such that $\alpha(x) \equiv \gamma_1(x)$, $\gamma_1(y) = \gamma_2(y)$, $\gamma_2(x) = \gamma_3(x)$, $\gamma_3(y) = \beta(y)$. Such γ_i are called* interpolants for α and β (with respect to x, y and z).

Notation 6.3.5 *We define equivalence relations on Θ for each $x \in \mathcal{L}$ by $\alpha \equiv_x \beta$ if and only if $\alpha(x) = \beta(x)$. For sequences p, q from Θ of length n and $x \in L$, we say $p \equiv_x q$ if $p(k) \equiv_x q(k)$ for every $k < n$. In general, we say an equivalence relation E on a set S is* larger *or* coarser *than another one $\hat{E}$ if for every $(\forall a, b \in S)(a \equiv_{\hat{E}} b \Rightarrow a \equiv_E b)$. Similarly, E is* finer *or* smaller *than $\hat{E}$ if $(\forall a, b \in S)(a \equiv_E b \Rightarrow a \equiv_{\hat{E}} b)$. With this ordering on equivalence relations the lub of E and $\hat{E}$ is simply their intersection. Their glb is the smallest equivalence class on S that contains their union. This is also the transitive closure of their union under the two relations.*

The conditions of Definition latrep 6.3.4 can now be restated in terms of these equivalence relations:

1. $\alpha \equiv_0 \beta$ for all α and β and so $\equiv_0$ is the largest congruence class identifying all elements.

2. If $x \leq y$ then $\alpha \equiv_y \beta$ implies $\alpha \equiv_x \beta$ for all α and β and so $\equiv_x$ is larger than $\equiv_y$.

3. If $x \not\leq_{\mathcal{L}} y$ then there are α and β such that $\alpha \equiv_y \beta$ but $\alpha \not\equiv_x \beta$ and so $\equiv_x$ is not larger than $\equiv_y$.

4. If $x \vee y = z$ and $\alpha \equiv_x \beta$ and $\alpha \equiv_y \beta$ then $\alpha \equiv_z \beta$ and so $\equiv_z$ is the glb of $\equiv_x$ and $\equiv_y$.

5. If $x \wedge y = z$ then there are $\gamma_1, \gamma_2, \gamma_3 \in \Theta$ such that $\alpha \equiv_x \gamma_1 \equiv_y \gamma_2 \equiv_x \gamma_3 \equiv_y \beta$. So $\equiv_z$ is certainly contained in the lub of $\equiv_x$ and $\equiv_y$. It is part of the theorem that we can arrange it so that chains of length three suffice to generate the entire transitive closure.

Thus a lattice table produces a representation by equivalence relations with the dual ordering. A reason for reversing the order is that $\mathcal{D}$ is only an uppersemilattice. So joins always exist and we want them to correspond to the simple operation on equivalence relations of intersection. On the other hand, meets do not always exist and they then correspond to lub on equivalence relations which requires work to construct.

We now prove our representation theorem in terms of lattice tables.

Theorem 6.3.6 (Representation Theorem) *If $\mathcal{L}$ is a recursive (in A) partial lattice with $0, 1$ then there is a uniformly recursive (in A) lattice table Θ for $\mathcal{L}$.*

??A versions of this result and proof are in Lerman ?? and Shore ??. This version is based on a Greenberg and Montalbán ??.

Proof. Define $\beta_{x,i}$ for $x, y \in L$, $i = 0, 1$ by

$$\beta_{x,0}(y) = \begin{cases} \langle x, 0\rangle \text{ if } y \neq 0 \\ 0 \text{ if } y = 0 \end{cases} \qquad \beta_{x,1}(y) = \begin{cases} \beta_{x,0}(y) \text{ if } y \leq x \\ \langle x, 1\rangle \text{ if } y \nleq x \end{cases}$$

The set of these $\beta_{x,i}$ satisfy (1), (2) and (4). We now want to sequentially close off under adding interpolants as required in (3) for each relevant instance . To do so, we have some dovetailing procedure which does the following. Consider $x \wedge y = z$ and $\alpha \equiv_z \beta$. We want to add $\gamma_1, \gamma_2, \gamma_3$ as required in (3) and preserve the truth of (1), (2) and (4) in the expanded set. If $x \leq y$ or $y \leq x$, it is easy to do so just using α and β. If not (i.e. $x \nleq y$ and $y \nleq x$), then choose new numbers a, b, c, d not used yet and put for $w \in L$

$$\gamma_1(w) = \begin{cases} \alpha(s) \text{ if } w \leq x \\ a \text{ if } w \nleq x \end{cases} \qquad \gamma_2(w) = \begin{cases} \gamma_1(w) \text{ if } w \leq y \\ b \text{ if } w \leq x \text{ and } w \nleq y \\ c \text{ otherwise} \end{cases} \qquad \gamma_3(w) = \begin{cases} \beta(w) \text{ if } w \leq y \\ a \text{ if } w \leq x \text{ and } w \nleq y \\ d \text{ otherwise} \end{cases}$$

This is a recursive procedure and can check that it works. (Exercise) ■

Now we can turn to the proof of our embedding theorem.

Proof (of Theorem latemb 6.3.1). We begin with a lattice table Θ for $\mathcal{P}$ recursive in $\mathcal{L}$. We define a notion of forcing $\mathcal{P}$ with elements $p \in \Theta^{<\omega}$, the natural ordering $p \leq_{\mathcal{P}} q$ if $p \supseteq q$ and the obvious choice of V. Our generics will then be maps $G : \mathbb{N} \to L$. Define, for $x \in L$, $G_x : \mathbb{N} \to \mathbb{N}$ by $G_x(n) = G(n)(x)$. The desired embedding is the given by $x \mapsto \deg(G_x)$. We use a sufficient amount of genericity to prove that this map really is an embedding that preserves all the required structure. We follow the numbering of clauses in Definition latrep 6.3.4.

1. $G_0(n) = 0$ for all n and so 0 is preserved by our embedding.

2. Suppose $x \leq_{\mathcal{L}} y$. We must show that $G_x \leq_T G_y$. Given n, want to compute $G_x(n) = G(n)(x)$. Find any $\alpha \in \Theta$ such that $\alpha(y) = G(n)(y) = G_y(n)$, i.e. $\alpha \equiv_y G(n)$. One exists because $G(n)$ is one such. As Θ is uniformly recursive we can search for it. Then since $x \leq_{\mathcal{L}} y$ and $G(n) \equiv_y \alpha$, by Definition latrep 6.3.4(2) we have that $G(n) \equiv_x \alpha$ so $G(n)(x) = \alpha(x) = G_x(n)$.

4 Suppose $x \vee y = z$. We must show that $G_z \equiv_T G_x \oplus G_y$. By the preservation of order, $G_z \geq_T G_x \oplus G_y$, so it suffices to compute $G_z(n) = G(n)(z)$ from $G_x(n)$ and $G_y(n)$. We search for an $\alpha \in \Theta$ such that $\alpha(x) = G(n)(x)$ and $\alpha(y) = G(n)(y)$, i.e. $\alpha \equiv_{x,y} G(n)$. There is one and we can find it as above. Now as $\alpha \equiv_{x,y} G(n)$, $\alpha \equiv_z G(n)$ by Definition latrep 6.3.4(3), so $\alpha(z) = G(n)(z)$.

We can also say something about the image of 1 under the embedding. Given n, $G_1(n) = G(n)(1)$ so $G_1 \equiv_T G$ since by Definition 6.3.4(2) the value of any $\alpha \in \Theta$ at 1 determines it uniquely and it can be determined uniformly recursively (in A). Thus the greatest degree in the embedding is the degree of the generic G ($G \oplus A$ if we send every x to $G_x \oplus A$).

Until this point, we have not used any genericity. We now turn to nonorder and infimum.

3 Suppose $x \not\leq y$. We want to prove that $\Phi_e^{G_y} \neq G_x$ for every e. Suppose that G is 1-generic (in A which we will generally omit repeating for brevity) and consider the sets

$$S_e = \{p \in \Theta^{<\omega} : \exists n \Phi_e^{p_y}(n) \downarrow \neq p_x(n)\}$$

where $p_x \in \omega^{<\omega}$ is defined in the obvious way by $p_x(m) = p(m)(x)$. $S_e \in \Sigma_1$ because given σ we can compute $p(n)(x)$ (since Θ is uniformly recursive). Therefore, the 1-genericity of G implies that there is a $p \in \mathcal{G} \cap S_e$ or there is a $p \in \mathcal{G}$ no extension of which is in S_e. Suppose $p \in \mathcal{G} \cap S_e$, then $\Phi_e^{G_y}(n) \neq G_x(n)$ as $p_y \subset G_y$ and $p_x \subset G_x$ so we're done. Otherwise, no extension of p is in S_e but, for the sake of a contradiction, $\Phi_e^{G_y} = G_x$. Let α and β be as in Definition 6.3.4(3) for x and y. By the obvious density of the sets $D_n = \{p | \exists m > n(p(m) = \alpha\}$ and the 1-genericity of $\mathcal{G}$, there is a $q \leq p$ and an $m > |p|$ such that $q(m) = \alpha$ and $q \in \mathcal{G}$. Moreover as $\Phi_e^{G_y}(m) \downarrow$ by our assumptions, we may also guarantee that $\Phi_e^{q_y}(m) \downarrow$ by simply choosing q as a long enough initial segment of G. Consider now the condition $\hat{q}$ such that $\hat{q}(k) = q(k)$ for $k \neq m$ and $\hat{q}(m) = \beta$. Our choice of α, β and q guarantees that $\hat{q} \leq p$, $q \equiv_y \hat{q}$ and $q \not\equiv_x \hat{q}$. Thus $\Phi_e^{q_y}(m) \downarrow = \Phi_e^{\hat{q}_y}(m) \downarrow$ but $q_x(m) \neq \hat{q}_x(m)$. So one of q and $\hat{q}$ is in S_e by definition for the desired contradiction.

5 Suppose that $x \wedge y = z$ and $\Phi_e^{G_x} = \Phi_e^{G_y} = D$. We want to prove that $D \leq_T G_z$. Now the assertion that $\Phi_e^{G_x}$ and $\Phi_e^{G_y}$ are total and equal is Π_2. So let us assume that $\mathcal{G}$ is 2-generic (in A) and so there is (by Theorem 6.2.10) a $p \in \mathcal{G}$ such that p forces this sentence. So for each n and $q \leq p$ there is an $r \leq q$ such that $r \Vdash \Phi_e^{G_x}(n) \downarrow = \Phi_e^{G_y}(n) \downarrow$. We now wish to compute $D(n)$ from G_z. As above, we can recursively find a $q \leq p$ such that $q \Vdash \Phi_e^{G_x}(n) \downarrow = \Phi_e^{G_y}(n) \downarrow$ and $q_z \subset G_z$ (since some initial segment of G does this). We claim that $\Phi_e^{q_x}(n) = D(n)$. To see this consider a $t \in \mathcal{G}$ such that $t \leq p$ and $t \Vdash \Phi_e^{G_x}(n) \downarrow = \Phi_e^{G_y}(n) \downarrow$. Necessarily, $\Phi_e^{t_x}(n) \downarrow = \Phi_e^{t_y}(n) \downarrow = D(n)$ and $t \equiv_z q$. By suitably lengthening t or q we may assume that they have the same length m. Let $l = |p| < m$. We now use both the interpolants guaranteed by Definition 6.3.4(5) and the fact that p forces $\Phi_e^{G_x}$ and $\Phi_e^{G_y}$ to be total and equal.

For each k with $l \leq k < m$ we choose interpolants $\gamma_{k,i}$ (for $i \in \{1, 2, 3\}$) between $q(k)$ and $t(k)$ as in Definition 6.3.4(5). We let $q_i(k) = p(k) = t(k)$ for $k < l$ and $q_i(k) = \gamma_{k,i}$ for $l \leq k < m$. We also let $q_0 = q$ and $q_4 = t$. So $q = q_0 \equiv_x q_1 \equiv_y q_2 \equiv_x q_3 \equiv_y q_4 = t$. We now extend the q_i in turn to make them force convergence at n but remain congruent modulo z. In fact, we make a single extension for all of them. By the fact that $p \Vdash \Phi_e^{G_x} = \Phi_e^{G_y}$ and $q_1 \leq p$, we can find an $r_1 = q_1\hat{\ }s_1$ such that $r_1 \Vdash \Phi_e^{G_x}(n)\downarrow = \Phi_e^{G_y}(n)\downarrow$. We now extend $q_2\hat{\ }s_1$ to $r_2 = q_2\hat{\ }s_1\hat{\ }s_2$ such that $r_2 \Vdash \Phi_e^{G_x}(n)\downarrow = \Phi_e^{G_y}(n)\downarrow$. Finally we extend $q_3\hat{\ }s_1\hat{\ }s_2$ to $r_3 = q_3\hat{\ }s_1\hat{\ }s_2\hat{\ }s_3$. Let $s = s_1\hat{\ }s_2\hat{\ }s_3$ and consider $q_i\hat{\ }s$ for $i \leq 4$. Looking at each successive pair we see by the alternating congruences (between x and y) that they all force the same equal values for $\Phi_e^{G_x}(n)$ and $\Phi_e^{G_y}(n)$. Thus, by transitivity of equality and permanence of computations under extension, $\Phi_e^{q_x}(n) = \Phi^{t_x}(n) = D(n)$ as required.

■

??Omit "in A" case in proof and say relativize as exercise to save having to ..?

By Theorem deffor 6.2.2, the embedding of $\mathcal{L}$ given by the generic G produced in Theorem emblat ?? can be taken to be into the degrees below the double jump of $\mathcal{L}$. We can improve this by a direct construction ?? or the following result.

latemb1gen **Exercise 6.3.7** *The above proof that infima are preserves used 2-genericity. Give a proof (Antonio??) that uses only 1-genericity. ??Hint: Suppose that $x \wedge y = z$ and $\Phi_e^{G_x} = \Phi_e^{G_y} = D$. Consider the Σ_1 sets $T_e = \{t \mid \exists n(\Phi_e^{t_x}(n)\downarrow \neq \Phi_e^{t_y}(n)\downarrow\}$ and $S_e = \{s : \exists n, \exists q, s_0, s_2, r$(of the same length) $\Phi_e^{q_x}(n)\downarrow = \Phi_i^{q_y}(n)\downarrow \neq \Phi_e^{r_x}(n)\downarrow = \Phi_i^{r_y}(n)\downarrow$ and $q \equiv_x s_0 \equiv_y s \equiv_x s_2 \equiv_y r$ so $q \equiv_z r\}$ restricted to the conditions extending a t witnessing the 1-genericity condition for T_e.*

Exercise 6.3.8 *If $\mathcal{L}$ is a recursive lattice with 0 and 1 then it can be embedded in both $\mathcal{D}(\leq \mathbf{0}')$ and $\mathcal{D}(\leq \mathbf{g})$ preserving both 0 and 1 for any 1-generic $\mathbf{g}$. (GM??)*

??Direct construction below $0'$ search for interpolants find them of condition with no extensions forcing convergence at a particular location??

Next, we disprove the homogeneity conjecture for $\mathcal{D}' = \langle \mathcal{D}, \leq_T, ' \rangle$. The conjecture, like that for $\mathcal{D}$, was based on the empirical fact that every theorem about the degrees or the degrees with the jump operator relativizes and so if true in $\mathcal{D}$ (or $\mathcal{D}'$) then it is true in $\mathcal{D}(\geq \mathbf{c})$ or $\mathcal{D}'(\geq \mathbf{c})$ for every $\mathbf{c}$. The conjectures asserted then that $\mathcal{D} \cong \mathcal{D}(\geq \mathbf{c})$ and even that $\mathcal{D}' \cong \mathcal{D}'(\geq \mathbf{c})$ for every degree $\mathbf{c}$.

jhomc **Theorem 6.3.9** *There is c such that $(\mathcal{D}, \leq, ') \not\cong (\mathcal{D}(\geq c), \leq, ')$.*

Corollary 6.3.10 *The homogeneity conjecture for $\mathcal{D}'$ does not hold.*

Proof. If it did, then $[0, 0''] \cong [c, c'']$. To find a contradiction, it's sufficient to find partial lattice recursive in c which cannot be embedded in $[0, 0'']$. (Using Exercise latemb1gen 6.3.7 one can replace $0''$ and c'' by $0'$ and c', respectively.)

There are continuum many finitely generated lattices (fact of lattice theory, true for ≥ 4 generators). But, only countably many finitely generated lattices can be embedded in $[0, 0'']$ (because the lattice embedded is determined by image of its generators). Choose $\mathcal{L}$ finitely generated but not embeddable in $[0, 0']$. $\mathcal{L}$ has some degree, say c. By theorem latemb 6.3.1 relativized to c, $\mathcal{L}$ is embeddable in $[c, c'']$. Thus $[0, 0''] \not\cong [c, c'']$ as required. ■

??Feiner, 1969 or so, gave Σ_1 presentable Boolean algebra which cannot be recursively presented. As a corollary to this and difficult but known initial segment results he gave the first proof of Theorem jhomc 6.3.9.??

??Put in when need it and can prove theorem?? For later applications, we would like to have specific more complicated lattices embedded below $0'$ than are given directly by Theorem latemb 6.3.1 or Exercise latemb1gen 6.3.7. To do so, we consider *effectively generated successor structures*. Explain/define w/o g_i then add?? Consider the generators $e_0, e_1, d_0, f_0, f_1, g_0, g_1$ with relations

$$(d_{2n} \vee e_0) \wedge f_1 = d_{2n+1} \qquad\qquad (d_{2n+1} \vee e_1) \wedge f_0 = d_{2n+2}.$$

So e_0, e_1, d_0, f_0, f_1 give lattice structure. g_0, g_1 will pick out the set. Given set S,

$$n \in S \Leftrightarrow d_n \leq g_0, g_1.$$

Thus, we code S into partial lattice $\mathcal{L}_S$. How complicated can S be? How hard is it to recover S from $\mathcal{L}_S$? If $\{d_n\}$ formed an independent set, then for every S, get g_0, g_1 and ideals generated by S are distinct. That is,

$$I_S = \{x : x \leq g_0, g_1\}.$$

To guarantee this, can write down axioms in terms of join, order that yield independence of $\{d_n\}$. Or, can add elements to the lattice: $\hat{d}_n$ above d_m for all $m \neq n$ and $\hat{d}_n \wedge d_n = 0$.

Suppose we have $\mathbf{e_0, e_1, d_0, f_0, f_1, g_0, g_1 \leq 0'}$. Given the lattice generated by them, how complicated is S? To answer these questions, we need to understand the complexity of the structure $\mathcal{D}(\leq 0')$. That is, how hard is it to compute the various lattice operations in $\mathcal{D}$? Φ_e^X being total is a Π_2^X question. To ask if $\Phi_e^X \leq_T \Phi_i^X$, means

$$\exists j(\Phi_j^{\Phi_i^X} = \Phi_e^X) \;\Leftrightarrow\; \exists j \forall n \exists s(\Phi_{j,s}^{\Phi_{i,s}^X}(n) = \Phi_{e,s}(X)(n)).$$

Hence, order is Σ_3^X. What about join? There is a recursive f such that

$$\Phi_e^X \vee \Phi_i^X = \Phi_j^X \;\Leftrightarrow\; \Phi_{f(e,i)}^X \leq_T \Phi_j^X \;\&\; \Phi_{f(e,i)}^X \geq_T \Phi_j^X).$$

So join is Σ_3 on indices. Finally, infs $\Phi_e^X \wedge \Phi_i^X = \Phi_j^X$ is Π_4.

Infima add another quantifier alternation. We may try to get around them: instead of d_1, consider cone below d_1 (which we can get using only order) excluding 0. To exclude 0, we need to say " not below 0". This is a Π_3 statement – better than Π_4 but still not great. Is there a positive way of saying that you're not equal to 0?

Start by defining positive Σ_1 formulas φ_n in $\leq, \vee$ such that $\varphi_n(x)$ iff $x \leq d_n$. By recursion on n,

$$\varphi_0(x) \equiv x = d_0; \qquad \varphi_{2n+1}(x) \equiv \exists y(\varphi_{2n}(y) \& x \leq (y \vee e_0), f_1);$$
$$\varphi_{2n+2}(x) \equiv \exists y(\varphi_{2n+1}(x) \& x \leq (y \vee e_1), f_0)$$

These φ pick out equivalence classes below each d_n. Is this enough to determine S? Suppose we have x such that $\varphi_n(x)$ and $x \leq g_0, g_1$. We still need to ensure that $x \neq 0$. Is there a way to say this with a positive formula? Add parameters p, q to the lattice with the properties

$$p \not\geq q; \qquad \forall n(p \vee d_n \geq q).$$

Then we can modify φ_n by adding $(x \vee p \geq q)$ to all of them.

The condition on independence of the d_n ($\hat{d_n} \wedge d_n = 0$) implies that $\neg\exists x(0 < x < d_n, d_m)$ for $n \neq m$. So,

$$\begin{aligned} 2n+1 \in S &\Leftrightarrow \exists x(\varphi_{2n+1}(x) \& x \leq g_0, g_1) \\ 2n+2 \in S &\Leftrightarrow \exists x(\varphi_{2n+2}(x) \& x \leq g_0, g_1). \end{aligned}$$

Therefore, if have a lattice $\mathcal{L}_S$ embeddable below degree of set X, then $S \in \Sigma_3^X$.

Notice that if leave out g_0, g_1, the partial lattice that remains is recursive. So, we know that it is embeddable below any 1-generic. Recall that in the Exact Pair theorem, the construction was recursive in $(\oplus A_i)'$. We will see that can build S recursive in $0'$ using these lattice embeddings.

Exercise 6.3.11 *Given $S \in \Sigma_3$, show that can embed $\mathcal{L}_S$ below $0'$, and probably below any 1-generic. (May be hard.)*

??Do now or later??

Chapter 7

The Theories of $\mathcal{D}$ and $\mathcal{D}(\leq \mathbf{0}')$

In the previous section, we talked about isomorphisms and embeddability issues. We need to consider more in order to understand theory of the degrees. We now approach theorems which say that the theories of (sets of sentences true in) $\mathcal{D}$ and $\mathcal{D}(\leq 0')$ are as complicated as possible. More precisely they are of the same Turing (even $1-1$) degree as true second and first order arithmetic, respectively.

7.1 Interpreting Structures

interp

??Explain interpreting one structure in another for first order structures. Example of $\mathcal{D}(\leq 0')$ in arithmetic. Then second order arithmetic and logic (on countable structures). Equivalence. Role of parameters, equivalence relations. So for $\mathcal{D}$ we need to code countable subsets and quantification over arbitrary relations on them. For $\mathcal{D}(\leq 0')$ will want to code arithmetic??

In the next section, we will show that we can code and quantify over all countable relations on $\mathcal{D}$ by quantifying over elements of $\mathcal{D}$. We use this coding to show that $Th(\mathcal{D}) \equiv_1 Th^2(\mathbb{N}, \leq, +, \times, 0, 1)$. Clearly, any sentence of $Th(\mathcal{D})$ can be interpreted in $Th^2(\mathbb{N}, \leq, +, \times, 0, 1)$ because the relation $X \leq_T Y$ is arithmetic. To interpret $Th^2(\mathbb{N}, \leq, +, \times, 0, 1)$ inside $Th(\mathcal{D})$, we need to encode the standard model of arithmetic. Such an encoding will be a set N of degrees, relations $R_\leq, R_+, R_\times$ on N, and degrees $n_0, n_1 \in N$ such that the axioms of Robinson arithmetic hold when

- quantification is over N, and
- $\leq, +, \times$ are interpreted as $R_\leq, R_+, R_\times$, and
- $0, 1$ are interpreted as n_0, n_1, and
- every nonempty subset of N has a least element.

Such encodings are definable from parameters, and we quantify over subsets of N by quantifying over degrees $\bar{p}$ defining countable sets. Thus, given a sentence φ of second order arithmetic, we test whether φ is true by asking if there is such a model $(n, R_{\leq}, R_+, R_{\times}, n_0, n_1)$ as above for which the translation of φ is true in $\mathcal{D}$.

We begin with $\mathcal{D}$ and coding countable subsets of pairwise incomparable degrees. To prove this, we will use Slaman-Woodin forcing. We will then show how to deal with arbitrary countable sets of, and relations on, degrees.

7.2 Slaman-Woodin Forcing and $Th(\mathcal{D})$

Let $\mathbf{S} = \{\mathbf{c}_i | i \in \mathbb{N}\}$ be a countable set of pairwise incomparable degrees. We want to make $\mathbf{S}$ definable in $\mathcal{D}$ from three parameters $\mathbf{c}$, $\mathbf{g}_0$ and $\mathbf{g}_1$. The definition will be that $\mathbf{S}$ is the set of minimal degrees $\mathbf{x} \leq \mathbf{c}$ such that $(\mathbf{x} \vee \mathbf{g}_0) \wedge (\mathbf{x} \vee \mathbf{g}_1) \neq \mathbf{x}$ in the strong sense that there is a $\mathbf{d} \leq \mathbf{x} \vee \mathbf{g}_0, \mathbf{x} \vee \mathbf{g}_1$ such that $\mathbf{d} \nleq \mathbf{x}$.

`SW` **Theorem 7.2.1** *For any set $S = \{C_0, C_1, \ldots, \}$ of pairwise Turing incomparable subsets of $\mathbb{N}$ let $C = \oplus C_i$. There are then G_0,G_1 and D_i such that, for every $i \in \mathbb{N}$ and $j < 2$,, $D_i \leq_T C_i \oplus G_j$ while $D_i \nleq_T C_i$. Moreover, the C_i are minimal with this property among sets recursive in C in the sense that for any $X \leq_T C$ for which there is a D such that $D \leq_T X \oplus G_j$ $(j < 2)$ but $D \nleq_T X$ there is an i such that $X \leq_T C_i$.*

Proof. Without loss of generality we may assume that each C_i is recursive in any of its infinite subsets: simply replace C_i by the set of binary stings σ such that $\sigma \subset C_i$. We take C to be $\oplus C_i$, the uniform join of the C_i. We build G_i as required by forcing in such a way as to also uniformly define the D_i from G_0 and C_i so that D_i will also be recursive in $G_1 \oplus C_i$ as well. We begin with the coding scheme that says how we compute the D_i.

Let $\{c_{i,0}, c_{i,1}, \ldots\}$ list C_i in increasing order. Our plan is that $D_i(n)$ will be $G_0(c_{i,n})$. To make sure that $D_i \leq_T G_1 \oplus C_i$ as well, we will guarantee that $G_0^{[i]} =^* G_1^{[i]}$. (Recall that $=^*$ means equality except possibly on a finite set.) We now turn to our notion of forcing $\mathcal{P}$.

The forcing conditions are p of triples the form $\langle p_0, p_1, F_p\rangle$ where $p_0, p_1 \in 2^{<\omega}$, $|p_0| = |p_1|$, and F_p is a finite subset of ω. We let the length of condition p be $|p| = |p_0| = |p_1|$. Refinement of the forcing conditions is defined by

$$\begin{aligned} p \leq q \;\Leftrightarrow\; & p_0 \supseteq q_0, p_1 \supseteq q_1, F_p \supseteq F_q, \text{ and} \\ & \text{if } i \in F_p \text{ and } |q| < \langle i, c_{i,n}\rangle \leq |p| \text{ then } p_0(\langle i, c_{i,n}\rangle) = p_1(\langle i, c_{i,n}\rangle). \end{aligned}$$

This is a finite notion of forcing with extension recursive in C. Our generic object defined from a filter $\mathcal{G}$ will be $G_0 \oplus G_1$ where $G_k = \cup\{p_k | p \in \mathcal{G}\}$. We use $\mathtt{G}_k$ in our language to mean the k^{th} coordinate the generic object. The function V is defined in the obvious way: $V(p) = p_0 \oplus p_1$. Note that $C \leq_T \mathcal{P}$ as well (Exercise) and so n-generic for $\mathcal{P}$ means generic for all Σ_n^C sets.

We call $\langle i, k\rangle$ a coding location for C_i if $k \in C_i$. The definition of extension above implies that extensions of p agree at the coding locations for C_i for $i \in F_p$. As the sets $\{p | n \in F_p\}$ are

Note that for any $\varphi \in \Sigma_0$, if $p \Vdash \varphi$ then $(p_0, p_1, \emptyset) \Vdash \varphi$. So if $q \leq p$ and $q \Vdash \psi$ for $\psi \in \Sigma_1$ then $(q_0, q_1, F_P) \Vdash \psi$ as well.

Suppose that $\mathcal{G}$ is 1-generic for $\mathcal{P}$. It is immediate from the definition of $\leq_{\mathcal{P}}$ and the density of the recursive (in $\mathcal{P}$) sets $\{p | i \in F_p\}$ that $G_0^{[i]}$ and $G_1^{[i]}$ differ on at most finitely many $n \in C_i$. (If $i \in F_p$ and $p \in \mathcal{G}$ then $G_0^{[i]}(m) = G_1^{[i]}(m)$ for $m \in C_i$ and $m > |p|$.) Thus $D_i \leq_T G_1 \oplus C_i$ as required.

We next show that $D_i \not\leq_T C_i$, that is $\Phi_e^{C_i} \neq D_i$ for each e. Suppose for the sake of a contradiction that $D_i = \Phi_e^{C_i}$ for some e. Consider the Σ_1^C set

$$S_{i,e} = \{p : \exists m (p_0(\langle i, c_{i,m}\rangle) \neq \Phi_e^{C_i}(m))\}.$$

Then S_i is dense because if $p \in P$ and m is such that $\langle i, c_{i,m}\rangle > |p|$ then define $q \leq p$ by $F_q = F_p$ and for $|p| \leq j \leq \langle i, c_{i,m}\rangle$ put $q_0(j) = q_1(j) = 1 - \Phi_e^{C_i}(m)$. So $q \in S_{i,e}$ and $q \leq p$ as desired. Thus, there is $p \in \mathcal{G} \cap S_{i,e}$ for which

$$D_i(m) = G_0(\langle i, c_{i,m}\rangle) = p_0(\langle i, c_{i,m}\rangle) \neq \Phi_e^{C_i}(m),$$

contradicting $D_i = \Phi_e^{C_i}$.

Now, we have to ensure minimality. In other words, we want to prove that if

$$\Phi_e^{X \oplus G_0} = \Phi_e^{X \oplus G_1} = D, \qquad X \leq_T C, \qquad D \not\leq_T X$$

then $C_k \leq_T X$ for some k. Consider the sentence φ that says that $\Phi_e^{X \oplus G_0}$ are $\Phi_e^{X \oplus G_1}$ total and equal. It is Π_2 in C (because $X \leq_T C$) and true of $G = G_0 \oplus G_1$. So, if we now assume that $\mathcal{G}$ is 2-generic, there is $p \in \mathcal{G}$ such that $p \Vdash \varphi$. Suppose first that $\neg \exists n\ (\exists \sigma \supseteq p_0)$ $(\exists \tau \supseteq p_0)[\Phi_e^{X \oplus \sigma}(n) \downarrow \neq \Phi_e^{X \oplus \tau}(n) \downarrow]$. Then D is computable from X: to compute $D(n)$ search for any $\sigma \supseteq p_0$ such that $\Phi_e^{X \oplus \sigma}(n) \downarrow$ and output this as the answer. There is such a $\sigma \subset G_0$ by the totality of $\Phi_e^{X \oplus G_0}$. Our assumption that there is no pair of extensions of p_0 that give two different answers implies that any such σ gives the answer $\Phi_e^{X \oplus G_0}(n) = D(n)$.

On the other hand, suppose there is such a splitting for n given by $p_0\hat{}\sigma$, $p_0\hat{}\tau$. By extending one of σ and τ if necessary, we may assume that $|\sigma| = |\tau|$. We claim that $p_0\hat{}\sigma$ and $p_0\hat{}\tau$ differ at a coding location $\langle k, c_{k,m}\rangle$ for some $k \in F_p$. Let τ' be such that

$$\Phi_e^{X \oplus (p_1\hat{}\tau\hat{}\tau')}(n) \downarrow = \Phi_e^{X \oplus (p_0\hat{}\tau\hat{}\tau')}(n) \downarrow.$$

There must be such a τ' as $(p_0\hat{}\tau, p_1\hat{}\tau, F_p) \leq p$ and so it has a further extension $q = (p_0\hat{}\tau\hat{}\rho_0, p_1\hat{}\tau\hat{}\rho_1, F_p)$ which forces $\Phi_e^{X \oplus G_0}(n) \downarrow = \Phi_e^{X \oplus G_1}(n) \downarrow$. Next consider $\hat{q} = (p_0\hat{}\tau\hat{}\rho_0, p_1\hat{}\tau\hat{}\rho_0, F_p) \leq p$. It also has an extension $(p_0\hat{}\tau\hat{}\rho_0\hat{}\mu_0, p_1\hat{}\tau\hat{}\rho_0\hat{}\mu_1, F_p) \Vdash$ $\Phi_e^{X \oplus G_0}(n) \downarrow = \Phi_e^{X \oplus G_1}(n) \downarrow$. It is now clear that $\tau' = \rho_0\hat{}\mu_1$ has the desired property. Next, consider the condition $q = (p_0\hat{}\sigma\hat{}\tau', p_1\hat{}\tau\hat{}\tau', F_p)$. Notice that $q \not\leq p$ because:

1. $\Phi_e^{X \oplus (p_0\hat{}\sigma)}(n) = \Phi_e^{X \oplus (p_0\hat{}\sigma\hat{}\tau')}(n)$ as $p_0\hat{}\sigma\hat{}\tau' \supseteq p_0\hat{}\sigma$.

2. $\Phi_i^{X \oplus (p_1\hat{}\tau\hat{}\tau')}(n) = \Phi_e^{X \oplus (p_0\hat{}\tau)}(n)$ by choice of τ', but

3. $\Phi_e^{X\oplus(p_0\hat{}\sigma)}(n) \neq \Phi_e^{X\oplus(p_0\hat{}\tau)}(n)$ because $n, p_0\hat{}\sigma, p_0\hat{}\tau$ were chosen to be splitting.

Hence, $\Phi_e^{X\oplus(p_0\hat{}\sigma\hat{}\tau')}(n) \neq \Phi_e^{X\oplus(p_1\hat{}\tau\hat{}\tau')}(n)$ and so q does not extend p. However, $p_0\hat{}\sigma\hat{}\tau' \supseteq p_0$ and $p_1\hat{}\tau\ \hat{}\tau' \supseteq p_1$, so it must be that $p_0\hat{}\sigma\hat{}\tau'$ and $p_1\hat{}\tau\ \hat{}\tau'$ differ at a coding location. Therefore, $p_0\hat{}\sigma$ and $p_0\hat{}\tau$ differ at a coding location $\langle k, n\rangle$ with $k \in F_p$.

We now show that there must be such $p_0\hat{}\sigma$ and $p_0\hat{}\tau$ which differ at only one number (which then must be a coding location $\langle k, n\rangle$ for some $k \in F_p$). Suppose σ, τ are strings as above with $|\sigma| = |\tau| = \ell$. Let $\sigma = \gamma_0^0, \gamma_1^0, \ldots, \gamma_z^0 = \tau$ be a list of strings in $\{0,1\}^\ell$ such that $\gamma_i^0, \gamma_{i+1}^0$ differ at only one number for each i. Let β be such that $\Phi_e^{X\oplus(p_0\hat{}\gamma_1^0\hat{}\beta)}(n)\downarrow$ (such a β exists by the same argument as before). Set $\gamma_i^1 = \gamma_i^0\hat{}\beta$ for each $0 \leq i \leq z$. Repeat this process for each $j \leq z$. At step $j+1$, let β be such that $\Phi_e^{X\oplus(p_0\hat{}\gamma_{j+1}^j\hat{}\beta)}(n)\downarrow$, and set $\gamma_i^{j+1} = \gamma_i^j\hat{}\beta$ for each $0 \leq i \leq z$. At the end, we have strings $\gamma_0^z, \gamma_1^0, \ldots, \gamma_z^z$ such that $\Phi_e^{X\oplus(p_0\hat{}\gamma_i^z)}(n)\downarrow$ for each i, and $p_0\hat{}\gamma_i^z, p_0\hat{}\gamma_{i+1}^z$ differ at only one number for each i. Since

$$\Phi_e^{X\oplus(p_0\hat{}\gamma_0^z)}(n) = \Phi_e^{X\oplus(p_0\hat{}\sigma)}(n) \neq \Phi_e^{X\oplus(p_0\hat{}\tau)}(n) = \Phi_e^{X\oplus(p_0\hat{}\gamma_z^z)}(n),$$

there must be an i for which $\Phi_e^{X\oplus(p_0\hat{}\gamma_i^z)}(n) \neq \Phi_e^{X\oplus(p_0\hat{}\gamma_{i+1}^z)}(n)$. The strings $p_0\hat{}\gamma_i^z, p_0\hat{}\gamma_{i+1}^z$ differ at only one number and it must be a coding location $\langle k, m\rangle$ for some $k \in F_p$ as required.

Next, we show that X can find infinitely many coding locations $\langle k, m\rangle$ for some fixed $k \in F_p$. Suppose we want to find such a location $\langle k, m\rangle$ with $m > M$. Search for strings $p_0\hat{}\sigma$ and $p_0\hat{}\tau$ that agree on the first M positions, differ at only one position, and satisfy $\Phi_e^{X\oplus(p_0\hat{}\sigma)}(n) \neq \Phi_e^{X\oplus(p_0\hat{}\tau)}(n)$. Such strings must exist because we could have started the above analysis at any condition $q \in \mathcal{G}$ with $q \leq p$ (so we can find such strings agreeing on arbitrarily long initial segments). The position at which $p_0\hat{}\sigma$ and $p_0\hat{}\tau$ differ must be a coding location bigger than M. Since F_p is finite infinitely many of these coding locations must be for the same k. so can be given to X as data, X can find infinitely many coding locations $\langle k, c_{k,m}\rangle$ for any fixed k. Hence, X can enumerate an infinite subset of C_k and so can compute a (perhaps smaller) infinite subset of C_k and hence all of C_k by our initial assumption on the C_i. ■

As 2-genericity sufficed for the proof of the theorem above , we can get the required $G_i \leq_T C''$ and,indeed with $(G_0 \oplus G_1)'' \equiv_T C''$. We see below (Theorem sw0 7.3.1 and Exercise sw1gen 7.3.3) that we can do better.

Now we work toward coding arbitrary countable relations on $\mathcal{D}$.

joinw1gen **Proposition 7.2.2** *If H is a Cohen 1-generic relative to C then for any $X, Y \leq C$ if $X \oplus H^{[i]} \leq Y \oplus H^{[j]}$ then $i = j$ and $X \leq Y$.*

Proof. Suppose that for some e, $X, Y \leq_T C$, $\Phi_e^{Y\oplus H^{[j]}} = X \oplus H^{[i]}$ and consider the set

$$S_e = \{\sigma \in 2^{<\omega} : \exists n \left(\Phi_e^{Y\oplus\sigma^{[j]}}(n)\downarrow \neq X \oplus \sigma^{[i]}(n)\right)\}.$$

$S_e \in \Sigma_1(C)$ so either there is $\sigma \in S_e \cap H$ or there is $\sigma \subset H$ no extension of which is in S_e. The first alternative clearly violates our assumption that $\Phi_e^{Y \oplus H^{[j]}} = X \oplus H^{[i]}$ and so there is a $\sigma \subset H$ such that $\tau \notin S_e$ for all $\tau \supseteq \sigma$. Let $n = |\sigma^{[i]}|$. If $i \neq j$ and there were $\beta \supseteq \sigma^{[j]}$ such that $\Phi_e^{Y \oplus \beta}(2n+1)\downarrow$, we could extend σ to τ such that $\tau^{[j]} = \beta$ and $\tau^{[i]}(n) = 1 - \Phi_e^{Y \oplus \beta}(2n+1)$ (as the value of $\tau^{[i]}(n)$ is independent of $\tau^{[j]}$. In this case, we have

$$\Phi_e^{Y \oplus \tau^{[j]}}(2n+1) \downarrow \neq \tau^{[i]}(n) = (X \oplus \tau^{[i]})(2n+1)$$

and so $\tau \in S_e$, contradicting our choice of σ. Therefore, there can be no $\beta \supseteq \sigma^{[j]}$ making $\Phi_e^{Y \oplus \beta}(2n+1)$ converge while $\Phi_e^{Y \oplus H^{[j]}}$ is total by assumption and $\sigma^{[j]} \subset H^{[j]}$ for a contradiction. Thus $i = j$.

Next, we show that $X \leq_T Y$. To compute $X(n)$ from Y, search for a $\tau \supseteq \sigma$ such that $\Phi_e^{Y \oplus \tau^{[j]}}(2n)$ converges (such a τ exists because $\Phi_e^{Y \oplus H^{[i]}}$ is total and $\sigma^{[j]} \subset H^{[j]}$). Then, as usual, we claim that $\Phi_e^{Y \oplus \tau^{[j]}} = (X \oplus \tau^{[i]})(2n) = X(n)$ for if not, $\tau \in S_e$ and extends σ for a contradiction. ■

Theorem 7.2.3 *Every countable relation $R(x_0, \dots, x_{n-1})$ on $\mathcal{D}$ is definable from parameters. Indeed for each n there is a formula $\varphi(x_0, \dots, x_{n-1}, \bar{y})$ with $\bar{y}$ of length some $k > 0$ (depending only on n) which includes the clauses that $x_i \leq y_0$ for each $i < n$ such that as $\bar{\mathbf{p}}$ ranges over all k-tuples of degrees, the sets of n-tuples of degrees $\{\bar{\mathbf{a}} | \mathcal{D} \vDash \varphi(\bar{\mathbf{a}}, \bar{\mathbf{p}})\}$ range over all countable n-ary relations on $\mathcal{D}$.*

Proof. We want to define R from parameters. As R is countable the degrees in its domain are countable and so we may assume they are all uniformly below some degree $\mathbf{c}$ ($\mathbf{c}$ is the degree of a set C which is a uniform upper bound for all the sets X of degrees in the domain of R) which we take to be our first parameter. Let H be Cohen 1-generic over $C \in \mathbf{c}$ and $\mathbf{h}_{i,j}$ be the degree of $H^{[\langle i,j \rangle]}$. Suppose $\{\mathbf{x}_j\}$ lists all the degrees $\leq \mathbf{c}$. We code R using the following countable sets of pairwise incomparable degrees.

$$\mathcal{H}_i = \{\mathbf{h}_{i,j} | j \in \mathbb{N}\} \text{ for } i < n$$

$$\mathcal{F}_i = \{\mathbf{x}_j \vee \mathbf{h}_{i,j} | j \in \mathbb{N}\} \text{ for } i < n$$

$$\mathcal{R} = \{\mathbf{h}_{0,j_0} \vee \mathbf{h}_{1,j_1} \vee \cdots \vee \mathbf{h}_{n-1,j_{n-1}} : R(\mathbf{x}_{j_0}, \mathbf{x}_{j_1}, \dots, \mathbf{x}_{j_{n-1}})\}$$

Each of these sets consists of pairwise incomparable degrees. The first and third by the fact (Exercise ??) that for a Cohen 1-generic H the sets $H^{[k]}$ form a very independent set. (So, for any finite A and B, $\vee\{\mathbf{x} | \mathbf{x} \in A\} \leq \vee\{\mathbf{x} | \mathbf{x} \in B\}$ if and only if $A \subseteq B$.) The elements of each $\mathcal{F}_i$ are pairwise incomparable by Proposition joinwigen ??. Our defining formula φ for R is now

$$\&_{i<n}(\mathbf{x}_i \leq \mathbf{c}) \;\&\; (\exists \mathbf{y}_i)_{i<n}(\mathbf{y}_i \in \mathcal{H}_i \;\&\; \&_{i<n}(\mathbf{x}_i \vee \mathbf{y}_i) \in \mathcal{F}_i \;\&\; \bigvee_{i<n} \mathbf{y}_i \in \mathcal{R})$$

where we understand membership in the sets $\mathcal{H}_i$, $\mathcal{F}_i$ and $\mathcal{R}$ as being defined by the appropriate parameters. The verification that this formula defines the relation is straightforward. If $R(\bar{\mathbf{x}})$ then every element of the sequence $\bar{\mathbf{x}}$ is below $\mathbf{c}$ and is therefore equal to an $\mathbf{x}_{j_i}$ (for $i < n$). The degrees $\mathbf{h}_{i,j_i} \in \mathcal{H}_i$ then are the witness $\mathbf{y}_i$ required in φ. In the other direction, if φ holds of any n-tuple then all its elements are below $\mathbf{c}$ and we need to consider the situation where $\varphi(\mathbf{x}_{j_0}, \ldots \mathbf{x}_{j_{n-1}})$ for some j_i, $i < n$. Let the required witnesses be $\mathbf{y}_i$. As $\mathbf{y}_i \in \mathcal{H}_i$ and $(\mathbf{x}_{j_i} \vee \mathbf{y}_i) \in \mathcal{F}_i$, $\mathbf{y}_i = h_{i,j}$. Then as $\bigvee_{i<n} \mathbf{y}_i \in \mathcal{R}$, $R(\mathbf{x}_{j_0}, \mathbf{x}_{j_1}, \ldots, \mathbf{x}_{j_{n-1}})$.

The assertions in the Theorem about the form of the required formulas φ are now immediate. ■

Our discussions of coding second order arithmetic in §$\overset{\texttt{interp}}{7.1}$ now show that we have precisely determined the complexity of the theory of the Turing degrees.

Theorem 7.2.4 *$Th(\mathcal{D}, \leq) \equiv_1 Th^2(\mathbb{N}, \leq, +, \times, 0, 1)$.*

Slaman Woodin "Definability in the Turing Degrees" IL. J. Math 30 (1986), 320-334
Odifreddi, Shore "Global Properties of Local Degree Structures" 1971 Bul. U. M. I.
Greenberg, Montalbán "Embedding and Encoding Below a 1-generic" JSL 2003

7.3 $Th(D \leq 0')$

We want to now improve our coding results so that they become applicable below $\mathbf{0}'$. We begin with the Slaman and Woodin coding of sets of pairwise incomparable degrees.

`sw0'` **Theorem 7.3.1** *For any set $S = \{C_0, C_1, \ldots, \}$ of pairwise Turing incomparable subsets of $\mathbb{N}$ let $C = \oplus C_i$. There are then G_0,$G_1 \leq_T C'$ and D_i such that, for every $i \in \mathbb{N}$ and $j < 2$,, $D_i \leq_T C_i \oplus G_j$ while $D_i \not\leq_T C_i$. Moreover, the C_i are minimal with this property among sets recursive in C in the sense that for any $X \leq_T C$ for which there is a D such that $D \leq_T X \oplus G_j$ $(j < 2)$ but $D \not\leq_T X$ there is an i such that $X \leq_T C_i$.*

Proof. We build $D_i \leq_T G_0 \oplus C_i, G_1 \oplus C_i$ such that $D_i \not\leq_T C_i$. The requirements for diagonalization here are:

$$P_{e,i} : \Phi_e^{C_i} \neq D_i.$$

Let $X_j = \Phi_j^C$. We also have requirements for minimality:

$$R_{e,,j} : \Phi_e^{G_0 \oplus X_j} = \Phi_e^{G_1 \oplus X_j} = D \Rightarrow D \leq_T X_j \text{ or } \exists i (C_i \leq_T X_j).$$

We list all the requirements as Q_s. We build G_0, G_1 by finite approximations $\gamma_{0,s}, \gamma_{1,s}$ of equal length. As before we let $D_i(m) = G_0(\langle i, c_{i,m}\rangle)$ where $\{c_{i,m}\}$ is enumeration of C_i in increasing order. So $D_i \leq_T G_0 \oplus C_i$. We guarantee that $D_i \leq_T G_1 \oplus C_i$ as before by making sure that, for each i, $G_0(\langle i, m\rangle \neq G_1(\langle i, m\rangle$ for at most finitely many

$m \in C_i$. In particular we institute a *rule for the construction* that when we act to satisfy requirement Q_n at stage s by extending the current values of γ_k ($k = 0, 1$) we require, for $i \leq n$, $\langle i, m\rangle \geq |\gamma_{0,s}| = |\gamma_{1,s}|$ and $m \in C_i$, that the extensions $\gamma_k^{'}$ are such that $\gamma_0'(\langle i, m\rangle) = \gamma_1'(\langle i, m\rangle)$. As we will act to satisfy any Q_n at most once, this rule guarantees that there are at most finitely many relevant differences between G_0 and G_1 for each i.

At stage s, if $Q_s = P_{e,i}$, we act to satisfy $P_{e,i}$. Choose m such that $\langle i, c_{i,m}\rangle \geq |\gamma_{0,s}|$. Ask if $\Phi_e^{C_i}(m)\downarrow$. If not, let $\gamma_{k,s+1} = \gamma_{k,s}$ for $k = 0, 1$. (As usual this satisfies $P_{e,i}$.) If it does converge, extend each of $\gamma_{0,s}, \gamma_{1,s}$ by same string σ to $\gamma_{0,s+1}, \gamma_{1,s+1}$ with $\gamma_{0,s+1}(\langle i, c_{i,m}\rangle) \neq \Phi_e^{C_i}(m)$. This also satisfies the requirement because $D_i(m) = G_0(\langle i, c_{i,m}\rangle)$ by definition and trivially obeys the rule of the construction.

Note that C' can answer the question $\Phi_e^{C_i}(m)\downarrow$, so this action is recursive in C'.

If $Q_s = R_{e,j}$, this stage will have a substage for each requirement $Q_n = R_{e',j'}$ with $n \leq s$ that has not yet been satisfied. For notational convenience we write γ_k for $\gamma_{k,s}$ in the description of our action at stage s. At the end of each substage we will define successive extensions $\gamma_{k,l}$ of γ_k satisfying the rule of the construction. We first try to satisfy $R_{e,j}$ (which, of course, we have not attempted to satisfy before). We ask if $\exists x \exists \sigma_k \supseteq \gamma_k$ which satisfy the rule of our construction and such that the $\sigma_k \oplus X$ e-split, i.e.

$$\Phi_e^{\sigma_0 \oplus X_j}(x)\downarrow \neq \Phi_e^{\sigma_1 \oplus X_j}(x)\downarrow .$$

(Note that when we are acting to satisfy any Q_n checking if extensions of the current values of γ_k satisfy the rule of the construction is recursive in $\oplus\{C_i | i \leq n\}$ and so uniformly recursive in C. Thus this question can be answered by C'. There is one subtlety here. We must be careful with what we mean by a computation from X_j as there is no list of all the sets recursive in C that is uniformly recursive in C. So what we mean here is that there is a computation of Φ_j^C providing long enough initial segment of X_j so as to make the desired computations at m converge. This makes the whole question one that is Σ_1^C and so recursive in C'.) If the answer is yes, choose as usual the first such extensions (in a uniform search recursive in C) as $\gamma_{0,0}, \gamma_{1,1}$. Note that we have now satisfied $R_{e,j}$. If the answer is no, ask if $\exists x \exists \sigma, \tau$ $((\gamma_0\hat{}\sigma \oplus X_j)|_e(\gamma_0\hat{}\tau \oplus X))$. (This question is also $\Sigma_1(C)$).

- If not, let $\gamma_{k,s,0} = \gamma_{k,s}$. Then, as usual, if $\Phi_e^{G_0 \oplus X_j}$ is total, it is recursive in X as we will have $G_0 \supseteq \gamma_{0,0}$. To calculate it at x, find any σ such that $\Phi_e^{\gamma_0\hat{}\sigma \oplus X_j}(x)\downarrow$. This computation must give right answer. So in this case we have also satisfied $R_{e,i,j}$.

- If so,we can find such σ and τ (recursively in C). We interpolate between σ, τ with strings $\sigma = \delta_0 = \delta_1, \ldots, \delta_z = \tau$ which differ successively at exactly one number. Ask if $\exists \sigma_1$ such that $\Phi_e^{\gamma_0\hat{}\delta_1\hat{}\sigma_1 \oplus X_j}(x)\downarrow$. If not, let $\gamma_{k,0} = \gamma_k\hat{}\delta_1$. Note that this extension satisfies the rule of the construction and that we have satisfied $R_{e,j}$ by guaranteeing that $\Phi_e^{G_0 \oplus X_j}(x)\uparrow$. If yes, consider $\delta_2\hat{}\sigma_1$ and ask again if there is a σ_2 such that $\Phi_e^{\delta_2\hat{}\sigma_1\hat{}\sigma_2 \oplus X_j}(x)\downarrow$. If not, let $\gamma_{k,0} = \delta_2\hat{}\sigma_1$ as before obeying the rule of the construction and satisfying $R_{e,j}$. If so, we continue on inductively through the δ_k.

- Eventually we either define $\gamma_{k,0}$ and satisfy $R_{e,j}$ or we find $\sigma_1, \ldots, \sigma_z$ such that $\Phi_e^{\gamma_0\hat{}\delta_l\hat{}\rho\oplus X_j}(x)\downarrow$ for every $l \leq z$ where $\rho = \sigma_1\hat{}\ldots\hat{}\sigma_z$. In the second case, we set $\gamma_{k,0} = \gamma_{k,s}$. This action does not satisfy $R_{e,i}$ but it demonstrates that there are $\hat{\sigma}$ and $\hat{\tau}$ which differ at exactly one number and for which $(\gamma_0\hat{}\hat{\sigma} \oplus X)|_e(\gamma_0\hat{}\hat{\tau} \oplus X)$. The point here is that, as $\Phi_e^{\gamma_0\hat{}\delta_0\hat{}\rho\oplus X_j}(x)\downarrow = \Phi_e^{\gamma_0\hat{}\sigma\oplus X_j}(x)\downarrow \neq \Phi_e^{\gamma_0\hat{}\tau\oplus X_j}(x)\downarrow = \Phi_e^{\gamma_0\hat{}\delta_z\hat{}\varepsilon\oplus X_j}(x)\downarrow$, there is an l such that $\Phi_e^{\gamma_0\hat{}\delta_l\hat{}\rho\oplus X_j}(x)\downarrow \neq \Phi_e^{\gamma_0\hat{}\delta_{l+1}\hat{}\rho\oplus X_j}(x)\downarrow$ while $\delta_l\hat{}\rho$ and $\delta_{l+1}\hat{}\rho$ differ at exactly one number. Now consider $\gamma_1\hat{}\hat{\sigma}$. If there is no μ such that $\Phi_e^{\gamma_1\hat{}\hat{\sigma}\hat{}\mu\oplus X_j}(x)\downarrow$ then we can again satisfy $R_{e,j}$ by setting $\gamma_{k,s,0} = \gamma_{k,s}\hat{}\hat{\sigma}$. If there is such a μ, we compare $\Phi_e^{\gamma_1\hat{}\hat{\sigma}\hat{}\mu\oplus X_j}(x)\downarrow$ with $\Phi_e^{\gamma_0\hat{}\hat{\sigma}\hat{}\mu\oplus X_j}(x)\downarrow$ and $\Phi_e^{\gamma_0\hat{}\hat{\tau}\hat{}\mu\oplus X_j}(x)\downarrow$. As the last two are different one of them must be different from the first. If $\Phi_e^{\gamma_1\hat{}\hat{\sigma}\hat{}\mu\oplus X_j}(x)\downarrow \neq \Phi_e^{\gamma_0\hat{}\hat{\sigma}\hat{}\mu\oplus X_j}(x)\downarrow$, we would contradict our assumption that the answer to our very first question was no as $\gamma_1\hat{}\hat{\sigma}\hat{}\mu$ and $\gamma_0\hat{}\hat{\sigma}\hat{}\mu$ certainly satisfy the rule of the construction. If $\Phi_e^{\gamma_1\hat{}\hat{\sigma}\hat{}\mu\oplus X_j}(x)\downarrow \neq \Phi_e^{\gamma_0\hat{}\hat{\tau}\hat{}\mu\oplus X_j}(x)\downarrow$, the only way we would not have the same contradiction is if the one point at which $\hat{\sigma}$ and $\hat{\tau}$ differ must be a coding location $\langle k, c_{k,m}\rangle$ with $k < s$. Thus the only was our actions at this stage do not satisfy $R_{\langle e,j\rangle}$ is if there are $\hat{\sigma}\hat{}\mu$ and $\hat{\tau}\hat{}\mu$ which differ at at exactly one point such that $(\gamma_1\hat{}\hat{\sigma}\hat{}\mu \oplus X_j)|_e\gamma_0\hat{}\hat{\tau}\hat{}\mu \oplus X_j$ and for any such $\hat{\sigma}$ and $\hat{\tau}$ the point of difference must be a coding location $\langle k, c_{k,m}\rangle$ with $k < s$.

- In this last case we set $\gamma_{0,0} = \gamma_0$ and $\gamma_{1,0} = \gamma_{1,s}$. In any event, we now proceed to extend $\gamma_{1,0}$ (and then γ_1) in the same way but attempting to satisfy each $Q_n = R_{e',j'}$ with $n < s$ that has not yet been satisfied. After some finite number of such attempts we have tried them all, satisfying some and for the others producing one more example of an x and two strings $\hat{\sigma}$ and $\hat{\tau}$ differing at one number only (after $|\gamma_0|$) such that $(\gamma_0\hat{}\hat{\sigma} \oplus X_{j'})|_e(\gamma_1\hat{}\hat{\tau} \oplus X_{j'})$ for each $\langle e', j'\rangle$ which we have not yet satisfied and a guarantee that any two such strings differ at a coding location $\langle k, c_{k,m}\rangle$ with $k < s$.

- At the end of this process we let $\gamma_{k,s+1}$ be the final extension of γ_k that we have produced.

We now claim that all the requirements are satisfied. It is immediate that $P_{e,i}$ is satisfied when we act for $Q_s = P_{e,i}$ at stage s. Consider any $R_{e,j} = Q_{s_0}$. If we ever act so as to satisfy it at some stage s of the construction, it is clearly satisfied and we never act for it again. As we violate the rule of the construction at some $\langle k, c_{k,m}\rangle$ only when we act to satisfy requirement Q_n for $n \leq k$ and we do so at most once for each n, $D_i \leq_T G_1 \oplus C_i$ as required.

Finally, suppose that the first requirement that we never act to satisfy during the construction is Q_n. It must be some $R_{e,j}$. Suppose that all requirements Q_r for $r < n$ have been satisfied by stage $s_0 > n$. At each stage $s > s_0$ with $Q_s = R_{e',j'}$ we attempt to satisfy $R_{e,j}$ as some substage of the construction. As we fail, there are $\Phi_{e'}^{\delta_0 \oplus X_{j'}}(x) \downarrow \neq \Phi_{e'}^{\delta_1 \oplus X_{j'}}(x) \downarrow$ with $\delta_k \supseteq \gamma_{k,s} \supseteq \gamma_{k,n}$ which differ at exactly one point and any such pair differ at a coding location $\langle k, c_{m,k} \rangle$ with $k \leq n$. Recursively in X_j we can then search for and find infinitely many extensions δ_k of $\gamma_{k,n}$ with this property with the points at which they differ becoming arbitrarily large (as $|\gamma_{k,s}|$ is clearly going to infinity). As there are only finitely many $k \leq n$, there must be one $k \leq n$ for which infinitely many of these δ_k differ at a point of the form $\langle k, z \rangle$ with infinitely many different z. As every such point is a coding location, recursively in X we can compute an infinite subset of C_k, so by our initial assumption that each C_i is recursive in everyone of its infinite subsets $C_k \leq_T X_j$ as required to satisfy $R_{e,j}$ in the end. ■

Exercise 7.3.2 *It is easy to show that the G_i of Theorem $\overset{\texttt{sw0'}}{7.3.1}$ can be made to have (or already have) jumps below C'. What about $(G_0 \oplus G_1)'$?*

sw1gen **Exercise 7.3.3** *With the notation as in Theorem $\overset{\texttt{sw}}{7.2.1}$ show that for any $\mathcal{G}$ 1-generic for $\mathcal{P}$, G_0 and G_1 have the properties required by the Theorem. Hint: Greenberg and Montalbán.*

??This step-by-step construction is the same as the forcing argument we saw before, but grittier and we gain a quantifier. This helps us to determine the true complexity of $Th(\mathcal{D}, \leq \mathbf{0}')$. We'd like to show $Th(\mathcal{D}, \leq \mathbf{0}') \equiv_m Th(\mathbb{N}, +, \times, \leq)$.

If $\oplus C_i = C$ is low (that is, $C' \equiv_T 0'$) then can get coding $< \mathbf{0}'$. By the Lattice Embedding Theorem (Theorem ??), get effective successor model with top element 1-generic so low. Then get parameters via Slaman-Woodin forcing which code $\{d_n\}$. Can also code relations of arithmetic on them.

If start with 1-generic and add more things which are 1-generic relative to it, or low relative to it, then stay 1-generic so keep everything below $\mathbf{0}'$. Thus, can code arithmetic below $\mathbf{0}'$. That is, can code set $\mathbb{N}$, and relations $+, \times, <$ so that satisfy Robinson arithmetic. How do we say that this is a standard model of arithmetic? We can't quantify over all subsets of $(\mathcal{D}, \leq \mathbf{0}')$ because there are only countably many codes but continuum many subsets. It is enough to show that the standard part has a code in the model. Using our coding below $\mathbf{0}'$, we can find parameters below $\mathbf{0}'$ which code the $\{\mathbf{d}_n\}$ of the successor model , and arithmetic on them. The ideal generated by $\{\mathbf{d}_n\}$ is Σ_3 if allow the top element of the lattice as a parameter (because it is effective successor model). So need to show that Σ_3 in C ideal has exact pair below $\mathbf{0}'$, where C is arbitrary complete RE degree below $\mathbf{0}'$.??

7.3.1 Definability in $\mathcal{D}(\leq \mathbf{0}')$

jump classes

invariant under double jump

Chapter 8

Domination Properties

8.1 Introduction

Notions and ideas.

Definition 8.1.1 *1. The function g* dominates *the function f ($f < g$) if, for all but finitely many x, $f(x) < g(x)$.*

2. The degree $\mathbf{g}$ dominates *the function f if some $g \in \mathbf{g}$ dominates f.*

3. The function g dominates *the degree* $\mathbf{f}$ *if g dominates every function $f \in \mathbf{f}$.*

4. The degree $\mathbf{g}$ dominates *the degree* $\mathbf{f}$ *if for every $f \in \mathbf{f}$ there is a $g \in \mathbf{g}$ which dominates f.*

*We also sometimes express these relations in the passive form saying, for example, that f is g-*dominated *or f is* $\mathbf{g}$-dominated *for the first two relations. A function g that dominates the degree* $\mathbf{0}$ *is called* dominant*.*

In the literature a degree $\mathbf{f}$ that is not $\mathbf{0}$-dominated (i.e. there is an $f \in \mathbf{f}$ which is not dominated by any recursive function) is, for historical reasons unrelated to our concerns, called *hyperimmune*. If $\mathbf{f}$ is not hyperimmune, i.e. it is $\mathbf{0}$-dominated, is also called *hyperimmune free*. For example, we will see that every $\mathbf{0} < \mathbf{a} < \mathbf{0}'$ is hyperimmune (Theorem delta2dom 8.2.3) while the minimal degrees constructed by Spector (spectormin §9.2) are hyperimmune free.

0domtt **Exercise 8.1.2** *Prove that if* $\mathbf{a}$ *is* $\mathbf{0}$-*dominated and $B \leq_T A \in \mathbf{a}$ then $B \leq_{tt} A$. So any* $\mathbf{0}$-*dominated Turing degree consists of exactly one tt (and so wtt) degree. Hint: if $B = \Phi_e^A$ then consider the function f such that $f(n) = \mu s(\Phi_{e,s}^{A\restriction s}(n)\downarrow)$.*

8.2 R.E. and Δ^0_2 degrees

redom **Theorem 8.2.1** *If $A >_T 0$ is r.e. then there is a function $m \equiv_T A$ which is not* **0**-*dominated, i.e. it is not dominated by any recursive function. Indeed, any function g which dominates m computes A.*

Proof. For A r.e., let A_s be the standard approximation to A at stage s.?? Let m be the least modulus function for this approximation: $m(x) = \mu s(\forall t \geq s)(A_s \upharpoonright x = A_t \upharpoonright x)$. For r.e. sets, the approximation changes its mind at most once and is correct in the limit, so $m(x)$ is also the $\mu s(A_s \upharpoonright x = A \upharpoonright x)$ and is clearly of the same degree as A. Moreover, if $g(x) \geq f(x)$ for almost all x, then $A \leq_T g$ as $A \upharpoonright x = A_{g(x)} \upharpoonright x$ for all but finitely many x. Thus, if $A >_T 0$, then f is not dominated by any recursive function and any g that dominates f computes A. ■

The Shoenfield limit lemma (Theorem limitlemma 4.3.9) gives us a recursive approximation $h(x, s)$ to any $A \in \Delta^0_2$ (or equivalently $A \leq_T 0'$). So the least modulus function m makes sense for such an approximation as well. So does the second version used in the above proof. Here we call it the *computation function:* $f(x) = \mu(s > x)(\forall y < x)(h(y, s) = A(y))$ (for technical reasons, we don't consider first few stages). It calculates the first stage after x at which the approximation is correct up to x. But, since we are no longer looking at r.e. sets, the approximation might change even after it's correct and the computation function f need not be the same as the least modulus m. The two functions may not be the same even up to degree.

Exercise 8.2.2 *Find an $A <_T 0'$ and an approximation $h(x, s)$ to A for which the least modulus function m computes $0'$. On the other hand, the computation function f for h is always of the same degree as A.*

We can, nonetheless extend Theorem redom 8.2.1 to all $A \in \Delta^0_2$.

delta2dom **Theorem 8.2.3** *If A is Δ^0_2, then there is an $f \equiv_T A$ which is not* **0**-*dominated. Indeed, any function g which dominates f computes* **a**.

Proof. By the Shoenfield limit lemma, there is a recursive $h(x, s)$ such that $\lim_{s\to\infty} h(x, s) = A(x)$. Let $f(x)$ be the computation function for this approximation. Suppose $f < g$. We claim that even though $h(z, s)$ may change at $z < x$ for $s > f(x)$, we can still compute A from g. Let s_0 be such that $(\forall m \geq s_0)(f(m) < g(m))$. To calculate $A(n)$ for $n > s_0$ find an $s > n$ such that $h(n, t)$ is constant for $t \in [g(s), gg(s)]$. Since $h(n, t)$ is eventually constant, such an s exists. Moreover, we can find it recursively in g: compute the intervals $[g(n+1), gg(n+1)], [g(n+2), gg(n+2)], [g(n+3), gg(n+3)], \ldots$ checking to see if h is constant on the intervals. By the clause that makes $f(x) > x$ in the definition of the computation function and our choice of s_0, $gg(s) > fg(s) > g(s)$, so the first $t > g(s)$ at which h is correct for all elements below $g(s)$ is in $[g(s), gg(s)]$. For this t, $h(n, t) = A(n)$. As we chose s so that the value of $h(n, t)$ is constant on this interval, $A(n) = h(n, t)$ for any $t \in [g(s), gg(s)]$ and we have computed A recursively in g as required. ■

Exercise 8.2.4 *What are the correct relativizations of the previous two Theorems?*

Exercise 8.2.5 *The above results can be extended by iterating the notions of "r.e. in" or more generally "Δ^0_2 in" as long as one includes the lower degrees. We say that A 1-REA if it is r.e. then we define n-REA by induction: A is $n+1$-REA if A is of the form $B \oplus W^B_e$ where B is n-REA. (REA stands for r.e. in and above.) Prove that any n-REA set A has an $f \equiv_T A$ such that any $g > f$ computes A. Do the same with Δ^0_2 replacing r.e. These results can be carried into the transfinite. Prove, for example, that $0^{(\omega)}$ has the same property.*

re1gen **Theorem 8.2.6** *If $A > 0$ is r.e. and $\mathcal{P}$ is a recursive notion of forcing then there is is 1-generic $\mathcal{G}$ such that (the corresponding) G is recursive in A.*

Proof. We will build a 1-generic sequence p_s recursive in A. Let $f \leq_T A$ be the least modulus function for A. The requirements are

$$R_e : \text{for some } s,\ p_s \in S_e \text{ or } (\forall q \leq p_s)(q \notin S_e), \text{ where } S_e \text{ is } e\text{th } \Sigma_1 \text{ set of conditions.}$$

At stage s, we have a condition p_s. Note that we are thinking of P as a subset of $\mathbb{N}$ and so have the natural ordering $\leq$ on its members (and all of $\mathbb{N}$) as well as the forcing ordering $\leq_{\mathcal{P}}$. We say that R_e has been declared satisfied by stage s if there is a p_n with $n \leq s$ such that $p_n \in S_{e.f(s)}$. Find the least $e < s$ such that P_e has not yet been declared satisfied and such that $(\exists q \leq_{\mathcal{P}} p_s)(q \leq f(s)\ \&\ q \in S_{e,f(s)})$. For this e, choose the least such q and put $\gamma_{s+1} = q$. If there is no such e, let $p_{s+1} = p_s$.

To verify that the construction succeeds, suppose for the sake of a contradiction that e_0 is least such that

$$\neg\exists s(p_s \in S_{e_0} \vee (\forall q \leq_{\mathcal{P}} p_s)(q \notin S_{e_0})).$$

Choose $s_0 > e_0$ such that $\forall i < e_0$ if there is a $p_s \in S_i$ then there is on with $s < s_0$ and $p_s \in S_{e,f(s_0)}$ (so by this stage we have already declared satisfied all higher priority requirements that are ever so declared). We claim that we can now recursively recover the entire construction and the values of $f(s)$ for $s \geq s_0$. As this would compute A recursively, we would have our desired contradiction. Consider what happens in the construction at each stage $s \geq s_0$ in turn. Suppose we have p_s. At stage s we look for the least $e < s$ such that $(\exists q \leq_{\mathcal{P}} p_s)(q \leq f(s)\ \&\ q \in S_{e,f(s)})$. There is no such $e < e_0$ by our choice of s_0. If e_0 itself were such an e, we would act for it and declare P_{e_0} to be satisfied, contrary to our choice of e_0. On the other hand, by our choice of e_0 there is a $q \leq_{\mathcal{P}} p_s$ with $q \in S_{e_0}$. We can find such a q recursively (because we know it exists). We did not find this q in the construction at stage s because either $q > f(s)$ or $q \in S_{e_0} - S_{e_0,f(s)}$. So we can now find a bound t on $f(s)$ by finding the stage at which q enters S_{e_0}. Given $t \geq f(s)$ we can calculate $f(s)$ as the least z such that $A_z \upharpoonright s = A_t \upharpoonright s$. Once we have $f(s)$ we can recursively determine what happened at stage s of the construction and in particular the value of p_{s+1}. Thus we can continue our recursive computation of $f(s)$ as claimed. ■

Relativizing Theorem $\overset{\text{re1gen}}{8.2.6}$ to C gives, for any C recursive notion of forcing $\mathcal{P}$, a $G \leq_T A$ which is C 1-generic for $\mathcal{P}$ for any $A >_T C$ which is r.e. in C.

Exercise 8.2.7 *The crucial property of the function f used in the above construction was that there is a uniformly recursive function computing $f(x)$ from any number greater than it. Prove that if there is a partial recursive $\varphi(x,s)$ such that $(\forall s \geq f(x))(\varphi(x,s) = f(x))$ then f is of r.e. degree.*

recohen **Corollary 8.2.8** *If $\mathbf{a} > \mathbf{0}$ is r.e. then there is Cohen 1-generic $G <_T A$ and so, for example, every countable partial order can be embedded in the degrees below $\mathbf{a}$.*

Similarly we have

Corollary 8.2.9 *If $\mathbf{a}$ is r.e. in $\mathbf{b}$ and strictly above it, then every partial lattice recursive in $\mathbf{b}$ can be embedded into $[\mathbf{b}, \mathbf{a})$.*

renomax **Corollary 8.2.10** *If $\mathbf{a}$ is r.e. then every maximal chain in $(\mathcal{D}(\leq \mathbf{a}), \leq_T)$ is infinite. In fact, there is no maximal element less than $\mathbf{a}$ in $(\mathcal{D}(\leq \mathbf{a}), \leq_T)$.*

Proof. Suppose $\mathbf{b} < \mathbf{a}$. Then $\mathbf{a}$ is r.e. in and strictly above $\mathbf{b}$. Relativizing Theorem re1gen 8.2.6 to a $B \in \mathbf{b}$ and using Cohen forcing gives us a $G \leq_T A$ which is Cohen 1-generic over B. So the degrees of $B \oplus G^{[i]}$ are in fact all between $\mathbf{b}$ and $\mathbf{a}$ and even independent. ■

Exercise 8.2.11 *Prove that every recursive lattice $\mathcal{L}$ with 0 and 1 can be embedded in $\mathcal{D}(\leq\mathbf{a})$ preserving 0 and 1 for any r.e. $\mathbf{a}$.??*

We now apply Theorem re1gen 8.2.6 to provide the missing way of identifying the standard parts of effective successor models coded below $0'$ that we need to calculate the complexity of $Th(\mathcal{D}(\leq\mathbf{0}'))$.

sigma3ideal **Theorem 8.2.12** *If $A >_T C$, A is r.e. in C and I is an ideal in $\mathcal{D}(\leq C)$ such that $W = \{e : \Phi_e^C \in I\} \in \Sigma_3^C$ then there is an exact pair G_0, G_1 for I below A.*

Proof. We provide a C-recursive notion of forcing $\mathcal{P}$ such that any 1-generic for $\mathcal{P}$ gives an exact pair for I and apply Theorem re1gen 8.2.6 relativized to C. The conditions of $\mathcal{P}$ are of the form $p = \langle p_0, p_1, F_p, n_p \rangle$ where $p_i \in 2^{<\omega}$, $|p_0| = |p_1| = |p|$, $F_p \in \omega^{<\omega}$, $n_p \in \omega$ such that

$$(\forall i \in \{0,1\})(\forall \langle e,x,y \rangle)(\exists^{\leq 1} \langle w,m \rangle)\left(\langle e,x,y,w,m \rangle \in p_i\right).$$

We define V as expected $V(p) = p_0 \oplus p_1$. So for a 1-generic $\mathcal{G}$, we have $G_i = \cup\{p_i | p \in \mathcal{G}\}$. If $e \in W$, we want Φ_e^C to be coded into G_i. The unusual restriction above on conditions in P suggests how we intend to do this coding. Since $W \in \Sigma_3^C$ we have a relation $R \leq_T C$ such that $e \in W \Leftrightarrow \exists x \forall y \exists z R(e,x,y,z)$. We denote the pairs of elements of W and their witness by $\hat{W} = \{\langle e,x \rangle : \forall y \exists z R(e,x,y,z)$. To calculate Φ_e^C for $e \in W$, our plan is to first choose an x such that $\langle e,x \rangle \in \hat{W}$. We then search for $\langle w,m \rangle$ such that $\langle e,x,y,w,m \rangle \in G_i$ and announce that $\Phi_e^C(y) = m$. The definition of P

guarantees that this procedure gives at most one answer. The definition of the partial order $\leq_{\mathcal{P}}$ below will guarantee that this procedure makes only finitely many mistakes for any 1-generic. Genericity will also guarantee that, when $\langle e, x\rangle \in \hat{W}$, it gives a total function.

The number n_p in our conditions acts as a bound for how far we can verify the Π_2 assertion that x is a witness that $e \in W$ (and so also that Φ^C_e is total). The set F_p will tell us for which $\langle e, x\rangle$ we can make no further mistakes in our coding of Φ^C_e into $G^{\langle e,x\rangle}_i$ when we extend p. With this intuition, we define extension in $\mathcal{P}$ by $q \leq_{\mathcal{P}} p$ iff

$$q_i \supseteq p_i, \qquad F_q \supseteq F_p, \qquad n_q \geq n_p,$$

and

$$\begin{aligned}(\forall i \quad &\in \quad \{0,1\})(\forall \langle e,x,y,w,m\rangle \in [|p|,|q|)(\langle e,x\rangle \in F_p \ \&\ \langle e,x,y,w,m\rangle \in q_i \\ &\rightarrow \quad \Phi^C_{e,n_q}(y) = m \ \&\ \forall y' \leq y \exists z \leq n_q\,(R(e,x,y',z))\end{aligned}$$

Note that $\mathcal{P}$ is recursive in C.

re1gen Suppose that G_0, G_1 are given by a C-1-generic sequence $\langle p_s\rangle \leq_T A$ as in Theorem 8.2.6 relativized to C. We claim that G_0, G_1 are an exact pair for I.

First assume that $\langle e, x\rangle \in \hat{W}$. We show that $\Phi^C_e \leq_T G_i$. As the sets $\{p|\,\langle e,x\rangle \in F_p\}$ are obviously dense in $\mathcal{P}$, there is an s such that $\langle e,x\rangle \in F_{p_s}$. For any $\langle e,x,y,w,m\rangle \in p_t$ with $t > s$, $\Phi^C_e(y) = m$ by definition and so as noted above, the prescribed search procedure which is recursive in G_i returns only correct answers for $y > |p_s|$. Next, we claim that for each $y > |p_s|$, $i \in \{0,1\}$ and $m = \Phi^C_e(y)$ the Σ^C_1 sets $S_{e,x,y,m,i} = \{r|\exists w(\langle e,x,y,w,m\rangle \in r_i\}$ are dense below p_s. This guarantees that $\langle p_t\rangle$ meets each of these sets and so the search procedures are total and correctly compute $\Phi^C_e(x)$ for all but finitely many x. To see that these sets are dense below p_s, consider any $q \leq p_s$ with no w such that $\langle e,x,y,w,m\rangle \in q_i$. Choose any $w > |q|$ and define an $r \leq_{\mathcal{P}} q$ by making $|r| = \big\langle e,x,y,w,\Phi^C_e(y)\rangle + 1\big\rangle$, $r_i = q_i \cup \{\langle e,x,y,w,\Phi^C_e(y)\rangle\}$ (i.e. we let them be 0 at other points below the length), $F_r = F_q$ and letting n_r be the least $n \geq n_q$ such that $\forall y' \leq y \exists z < n(R(e,x,y',z)\ \&\ \Phi^C_{e,n}(y)\downarrow)$ (one such exists since we are assuming that $\langle e,x\rangle \in \hat{W}$). Then $r {\leq_{\mathcal{P}}} q$ and $r \in S_{e,x,y,m,i}$ as desired.

We next want to deal with the minimality conditions associated with the G_i being an exact pair for I. Suppose then that $\Phi^{G_0}_e = \Phi^{G_1}_e = D$ is total. We want to prove that $D \leq \oplus\{\Phi^C_e : e \in F\}$ for some finite $F \subset W$. Consider the Σ_1 set S_e of conditions p:

$$S_e = \{p : \exists n\,(\Phi^{p_0}_e(n)\downarrow \neq \Phi^{p_1}_e(n))\downarrow\}$$

By our assumption there is no $p_s \in S_e$ so we have a $p_s = p$ such that $\forall q \leq_{\mathcal{P}} p(q \notin S_e)$. We claim that $D \leq \oplus\{\Phi^C_e : \langle e,x\rangle \in F_p \cap \hat{W}\}$. For every $\langle e,x\rangle \in F_p \setminus \hat{W}$, let $y(e,x)$ be the least y such that $\neg\forall y' \leq y \exists z R(e,x,y',z) \vee \Phi^C_e(y)\uparrow$. It is clear that there is no $q \leq_{\mathcal{P}} p$ with any $\langle e,x,y,w,m\rangle \in q_i$ for $\langle e,x\rangle \in F_p \setminus \hat{W}$ and $y \geq y(e,x)$. Choose $q \leq_{\mathcal{P}} p$ in $\langle p_s\rangle$ so that it has the maximal number of y's with some $\langle e,x,y,w,m\rangle \in q_i$ for

$y < y(e,x)$ and $i \in \{0,1\}$. To compute $D(y)$ for $y > |q|$, we find a $t \in \mathcal{P}$ such that $t_i \supseteq q_i$, $\Phi_e^{t_0}(y)\downarrow = \Phi_e^{t_1}(y)\downarrow$, no elements not in q_i are added into t_i in columns $\langle e,x\rangle \in F_p \backslash \hat{W}$ and for any $\langle e,x,y,w,m\rangle \in t_i$ with $\langle e,x\rangle \in F_p \cap \hat{W}$, $\Phi_e^C(y) = m$. Such an extension exists because $\Phi_e^{G_0}(y)\downarrow = \Phi_e^{G_1}(y)\downarrow$ and by the maximality property of q and the definition of $\leq_{\mathcal{P}}$, $G_i^{[\langle e,x\rangle]} = q_i^{[\langle e,x\rangle]}$ for $\langle e,x\rangle \in F_p \setminus \hat{W}$ and so there is such a $\hat{t} \in \langle p_s\rangle$. Finding one such t is clearly recursive in $\oplus\{\Phi_e^C : \langle e,x\rangle \in F_p \cap \hat{W}\}$. Thus we only need to show that any such t_i provide the right answer. If one such gave an answer different than that given by $\hat{t}$ (and so G_0 and G_1) then $\langle t_0, \hat{t}_1, F_p, n\rangle$ (where $n \geq n_q$ is large enough so that $\Phi_{e,n}^C(y)\downarrow$ for every $\langle e,x,y,w,m\rangle$ in t_0 or $\hat{t}_1$ with $\langle e,x\rangle \in F_p \cap \hat{W}$) would be an extension of p in S_e for the desired contradiction. ■

8.3 High and $\overline{GL}_2$ degrees

We now look at stronger domination properties and their relation to the jump classes $\mathbf{H}_1$ and $\bar{\mathbf{L}}_2$ below $\mathbf{0}'$ and their generalizations. Recall from §4.6 that for $\mathbf{a} \leq \mathbf{0}'$, $\mathbf{a} \in \mathbf{H}_1 \Leftrightarrow \mathbf{a}' = \mathbf{0}''$; $\mathbf{a} \in \mathbf{L}_2 \Leftrightarrow \mathbf{a}'' = \mathbf{0}''$. For degrees $\mathbf{a}$ not necessarily below $\mathbf{0}'$, $\mathbf{a} \in \mathbf{GL}_2 \Leftrightarrow (\mathbf{a} \vee \mathbf{0}')' = \mathbf{a}''$; $\mathbf{a} \in \mathbf{GH}_1 \Leftrightarrow \mathbf{a}' = (\mathbf{a} \vee \mathbf{0}')'$. It is also common to say that $\mathbf{a}$ is *high* if $\mathbf{a}' \geq \mathbf{0}''$. As it turns out these are the degrees of dominant functions. Of course, $\mathbf{a} \in \overline{\mathbf{GL}}_2$ means that $\mathbf{a} \notin \mathbf{GL}_2$.

Let's begin by showing that there is there a dominant function. In fact, if $\mathcal{C}$ is any countable class of functions $\{f_i\}$ then there is function f which dominates all the f_i. For example, put $f(x) = \max\{f_i(x) : i < x\} + 1$. This construction requires a uniform list of all the functions f_i. For the recursive functions we know that $0''$ can compute such a list. Indeed, $Tot = \{e : \Phi_e \text{ total}\} \equiv_T 0''$ (Exercise 4.5.4) and so there is a sequence f_i uniformly computable from $0''$ which then computes a dominant function as described. We can do better than this and avoid using totality. If $f(x) = \max\{\Phi_e(x) : e < x \ \&\ \Phi_e(x)\downarrow\}$ then $f \leq_T 0'$ and is also clearly dominant. We can even do a bit better and get away with functions of high degree.

Theorem 8.3.1 (Martin's High Domination Theorem) *A set A computes a dominant function f if and only if $0'' \leq_T A'$.*

Proof. Suppose first that $0'' \leq_T A'$. By the Shoenfield limit lemma (Theorem 4.3.9) and the fact that $Tot \leq_T 0''$, there is an $h \leq_T A$ with $\lim_{s\to\infty} h(e,s) = Tot(e)$. We want to compute a function f recursively in A such that, for every e for which Φ_e is total, $f(x)$ is larger than $\Phi_e(x)$ with at most finitely many exceptions. Any such f will be dominant. To compute $f(x)$ we compute, for each $e < x$, both $\Phi_{e,t}(x)$ and $h(e,t)$ for $t \geq x$ until either the first one converges, say to y_e, or $h(e,t) = 0$. As if Φ_e is not total, $\lim h(e,t) = 0$, one of these outcomes must happen. We set $f(x)$ to be one more than the maximum of all the y_e so computed for $e < x$. Note that $f \leq_T h \leq_T A$. It remains to verify that if Φ_e is total then $\Phi_e < f$. By our choice of h, $\exists s_0(\forall s \geq s_0)(h(e,s) = 1)$.

So for $x > s_0$ when we calculate $f(x)$ we always find a t such that $\Phi_{e,t}(x)\downarrow = y_e$ and so $f(x) > \Phi_e(x)$ for all $x > s_0$.

For the other directions, suppose we have a dominant f. As Tot is Π^0_2 and computes $0''$, it suffices to show that it is also $\Sigma_2(f)$ as it would then be $\Delta_2(f)$ and so recursive in f'. We claim that

$$\forall x \exists s \Phi_{e,s}(x)\downarrow \;\Leftrightarrow\; \exists c \forall x \Phi_{e,f(x)+c}(x)\downarrow .$$

Suppose Φ_e is total (if not, then of course both conditions fail). Let $k(x) = \mu s \Phi_{k,s}(x)\downarrow$. Then k is recursive (because we know that$\forall x \Phi_e(x)\downarrow$). By hypothesis, f dominates k. Thus, the right hand side holds. This is a $\Sigma_2(f)$ formula as desired. ■

Now a look at the definitions shows that for $\mathbf{a} \leq_T \mathbf{0}'$, $\mathbf{a} \notin \mathbf{L}_2$ is equivalent to $\mathbf{0}'$ not being high relative to $\mathbf{a}$. Relativizing Theorem $\overset{\texttt{martin}}{8.3.1}$ to an $\mathbf{a} \leq_T \mathbf{0}'$ we see that $\mathbf{a} \notin \mathbf{L}_2$ if and only if no $f \leq_T 0'$ dominates every (total) function recursive in A. We can then handle $\overline{\mathbf{GL}}_2$ by relativizing to $\mathbf{a} \vee \mathbf{0}'$ to prove the following:

Proposition 8.3.2 *A set $A \leq_T 0'$ has degree in $\overline{\mathbf{L}}_2$ if and only if $(\forall g \leq_T 0')(\exists f \leq_T A)(f \not< g)$. An arbitrary set A has degree in $\overline{\mathbf{GL}}_2$ if and only if $(\forall g \leq_T A \vee 0')(\exists f \leq_T A)(f \not< g)$.*

This says that, while sets that are not high do not compute dominant functions, if they are not too low they compute functions which are not dominated by any recursive function. This suffices for many applications.

$\boxed{\texttt{gl21gen}}$ **Theorem 8.3.3** *If $A \notin GL_2$ then for any recursive notion of forcing $\mathcal{P}$ there is 1-generic $G \leq_T A$.*

Proof. For any $g \leq_T A \vee 0'$, there is an $f \leq_T A$ not dominated by g. Without loss of generality we may take f to be strictly increasing. We first construct the function g that we want and then using the associated f, we construct a 1-generic sequence p_s recursively in f (and so A). We again make use of the natural order $\leq$ on $P \subseteq \mathbb{N}$.

Let S_e list the Σ_1 subsets of P. As usual, we declare S_e to be satisfied at s if $(\exists n \leq s)(p_n \in S_{e,s})$. We define g by recursion using $0'$. Given $g(s)$, we want to determine $g(s+1)$. For each condition $p \leq g(s)+1$, ask $0'$ if $(\exists q \leq_{\mathcal{P}} p)(q \in S_e)$ for each $e \leq g(s)+1$. If such an extension exists, let x_e be the least x such that $(\exists q \leq_{\mathcal{P}} p)(q \leq x \;\&\; q \in S_{e,x})$. Put $g(s+1) = \max\{x_e | e \leq g(s)+1\}$.

We cannot use g itself in the construction of the desired 1-generic because want $G \leq_T A$. But, since $g \leq_T A \vee 0'$, we can use an increasing $f \leq_T A$ not dominated by g. The construction of G will be recursive in f (hence in A). At stage s, we have finite a condition p_s. For each $e \leq s$ not declared satisfied at s, see if $(\exists q \leq_{\mathcal{P}} p_s)(q < f(s+1) \;\&\; q \in S_{e,f(s+1)})$. If so, take the smallest such q for the least such e and let it be p_{s+1}. If not, $p_{s+1} = p_s$. The construction is recursive in f, hence in A. Thus $\langle p_s \rangle \leq_T A$ and the associated generic $G \leq_T A$ as well. Note that $p_s \leq f(s)$ by induction. Indeed $p_s \leq g(s)$ as well because $g(s)$ gives a bound on the witness required in the definition of p_s.

To verify that G is 1-generic suppose, for the sake of a contradiction, that there is a least e_0 such that

$$\neg\exists s(p_s \in S_{e_0} \vee (\forall p \leq_{\mathcal{P}} p_s)(p \notin S_{e_0})).$$

Choose s_0 such that $(\forall i < e_0)(\exists s)(S_i$ is declared satisfied at $s) \;\rightarrow\; S_i$ is declared satisfied at $s_0)$. Consider any $s > s_0$ at which $f(s+1) > g(s+1)$. By our choice of e_0, there is a $q \leq_{\mathcal{P}} p_s$ such that $q \in S_{e_0}$. Moreover, as $p_s \leq g(s)$, by definition of g there is one $\leq g(s{+}1)$ such that it belongs to $S_{e_0, g(s+1)}$ as well. By our choice of s, $q \leq g(s{+}1) < f(s{+}1)$. Thus at stage $s+1$, we would act to extend p_s to a $p_{s+1} \in S_{e_0}$ for the desired contradiction. ■

cohenanr **Remark 8.3.4** *The function g we used in the above proof was actually recursive in $0'$. In fact, for Cohen forcing $g \leq_{wtt} 0'$. Thus we used the weaker property that for every function $g \leq_{wtt} 0'$ there is an $f \leq_T A$ not dominated by g. This property is called array non-recursiveness and is discussed in the next section.*

As with r.e. degrees, having a 1-generic below a degree $\mathbf{a} \notin \mathbf{GL}_2$ provides a lot of information about the degrees below $\mathbf{a}$. For example, as in Corollary recohen 8.2.8, we can embed every countable partial order below any $\mathbf{a} \notin \mathbf{GL}_2$. It is tempting to think that we could also prove the analog of Corollary renomax 8.2.10 that every maximal chain in the degrees below $\mathbf{a}$ is infinite. This is true for $\mathbf{a} < \mathbf{0}'$ (Exercise ??) but was an open question in Lerman [1983]. Cai ?? has now proven that it is not true. There are $\mathbf{a} \notin \mathbf{GL}_2$ which are the tops of a maximal chain of length three.

Exercise 8.3.5 *Prove that if $\mathbf{a} \in \mathbf{L}_2$ then any maximal chain in the degrees below $\mathbf{a}$ is infinite.*

On the other hand, we can say quite a bit about the degrees above $\mathbf{a}$ as well when $\mathbf{a} \notin \mathbf{GL}_2$ that is not true of arbitrary r.e. degrees.

Definition 8.3.6 *A degree $\mathbf{a}$ has the* cupping property *if $(\forall \mathbf{c} > \mathbf{a})(\exists \mathbf{b} < \mathbf{c})(\mathbf{a} \vee \mathbf{b} = \mathbf{c})$.*

Theorem 8.3.7 *If $\mathbf{a} \in \overline{\mathbf{GL}}_2$ then $\mathbf{a}$ has the cupping property. Indeed, if $A \notin GL_2$ and $C >_T A$ then there is $G \not\geq_T A$ such that $A \vee G \equiv_T C$ and G is Cohen 1-generic.*

Proof. We need to add requirements $R_e : \Phi_e^G \neq A$ to the proof of Theorem gl21gen 8.3.3 for Cohen forcing (making all the requirements into a single list Q_e) and code C into G as well (so as to be recoverable from $A \oplus G$). In the definition of $g(s+1)$ in that proof, for each $p \leq g(s)+1$ look as well for $q_0, q_1 \supseteq p$ and x such that $q_0|_e q_1$. Then make $g(s{+}1)$ also bound the least such extensions τ_0, τ_1 for each $e, p \leq g(s)+1$ for which such extensions exist.

Again choose $f \leq_T A$ strictly increasing and not dominated by g. The construction is done recursively in $f \oplus C$. At stage s we have p_s and we look for the least e such that Q_e has not yet been declared satisfied and for which there is either a $q \leq_{\mathcal{P}} p$ with $q \leq f(s{+}1)$ that would satisfy Q_e as before if it is an S_i or $q_0, q_1 \supseteq p_s$ with $q_i \leq f(s+1)$ such that

$q_0|_e q_1$ if $Q_e = R_i$. Let e be the least for which there are such extensions. If $Q_e = S_i$ choose q as before. If it is R_i Let q be the q_j such that $\Phi_e^{q_j}(x) \downarrow \neq A(x)$. We then let $p_{s+1} = q\hat{\ }C(s)$ and declare Q_e to be satisfied. If there is no such e, we let $p_{s+1} = p_s\hat{\ }C(s)$. Note that $p_{s+1} \leq f(s+1)+1$ (the extra 1 comes from appending $C(s)$).

Since the construction is recursive in $f \oplus C$ and $f \leq_T A \leq_T C$, we have $G \leq_T C$. But, $C \leq_T \langle p_s \rangle$ because $C(s) = p_{s+1}(|p_{s+1}|)$. However, $\langle p_s \rangle \leq_T A \vee G$ because $f \leq_T A$ tells how to compute each stage from the given p_s to the choice of q. Then G tells us the last extra bit at end of p_{s+1}.

To verify that G has the other required properties suppose e_0 is least such that Q_e fails. Assume that by stage s_0 we have declared all requirements with $e' < e_0$ which will ever be declared satisfied to be satisfied. Consider a stage $s > s_0$ at which $f(s+1) > g(s+1)$. If $Q_e = S_i$ then we argue as in the previous theorem. If $Q_e = R_i$ and there were any $q_0, q_1 \supseteq p_s$ with $q_0|_e q_1$ then would have taken one of them as our q and declared $Q_e = R_i$ to be satisfied contrary to our choice of e_0. On the other hand, if there are no such extensions, then as usual Φ_e^G is recursive if total and so R_i would also succeed contrary to our assumption. ■

Remark 8.3.8 *Not every r.e. degree has the cupping property [??].*

For other results about $\mathbf{GL}_2$ degrees it is useful to strengthen Theorem gl21gen 8.3.3 to deal with notions of forcing recursive in A rather than just recursive ones.

gl2genseq **Theorem 8.3.9** *For $A \in \overline{GL}_2$, given an A recursive notion of forcing $\mathcal{P}$ and a sequence D_n of dense sets (including the sets $\{p|\ |V(p)| > m\}$ for each m) uniformly recursive in $A \vee 0'$ there is a generic sequence $\langle p_s \rangle \leq_T A$ meeting all the D_n. Of course, the generic G associated with the sequence is recursive in A as well.*

Proof. Let m_K be the least modulus function for $K = 0'$ and let $\Psi_n^{A\oplus K} = D_n$, i.e. the Ψ_n uniformly compute membership in D_n. We define $g \leq_T A \vee 0'$ by recursion. Given $g(s)$ we find, for each $p, n \leq g(s)+1$ the least q such that $q \leq_P p$ and $q \in D_n$. We then find the use u (from $A \oplus K$) needed to compute Ψ_n at each number less than or equal to any of these q. We then let $g(s+1)$ be the least number larger than q, u and $m_K(u)$ for all of these q and u as well as $m_K(g(s)+1)$. As $g \leq_T A \vee 0'$ and $A \in \overline{GL}_2$ there is an increasing $f \leq_T A$ not dominated by g.

We construct the sequence $\langle p_s \rangle$ recursively in $f \leq_T A$. At stage s we have p_s. Our plan is to satisfy the requirement of meeting D_n for the least n for which we do not seem to have done so yet and for which we can find an appropriate extension of p_s when we restrict our search to $q \leq f(s+1)$ as well as our use of $0'$ to what we have at stage $f(s+1)$. More formally, we determine (recursively in A) for which D_n ($n \leq s$) there is a $t \leq s$ such that $\Psi_n^{(A\oplus K_{f(s+1)})\restriction f(s+1)}(p_t) = 1$. Among the other $n \leq s$, we search (again recursively in A) for one such that $(\exists q \leq_{\mathcal{P}} p_s)(q \leq f(s+1)\ \&\ \Psi_n^{(A\oplus K_{f(s+1)})\restriction f(s+1)}(p_t) = 1)$. If there is one we act for the least such n by letting p_{s+1} be the least such q for this n. If not, let $p_{s+1} = p_s$. Note that $p_{s+1} \leq f(s+1)$ by the restriction on the search space and $p_{s+1} \leq g(s+1)$ as well since $g(s+1)$ also bounds the least witness by the definition of g.

We now claim that for each n there is a $p_s \in D_n$. If not, suppose, for the sake of a contradiction, that n is least counterexample. Choose s_0 such that for all $m < n$ there is $t < s_0$ such that $p_t \in D_m$ and indeed such that $\Psi_m^{(A \oplus K_{s_0})\upharpoonright s_0}(p_t) = 1$ and $K_{s_0} \upharpoonright u = K \upharpoonright u$ where u is the use of this computation of Ψ_m at p_t. Thus, by construction, we will never act for $m < n$ after s_0. As g does not dominate f we may choose an $s > s_0$ with $f(s+1) > g(s+1)$. At stage s we have p_s and $p_t \notin D_n$ for all $t \leq s$ in the sense required, i.e. $\Psi_n^{(A \oplus K_{f(s+1)})\upharpoonright f(s+1)}(p_t) = 0$ since any computation of this form gives the correct answer by our definition of $g(s+1)$ and the fact that $f(s+1) > g(s+1)$. There is a $q \leq_{\mathcal{P}} p_s$ with $q \in D_n$ and the least such is less than $f(s+1)$ and $\Psi_n^{(A \oplus K_{f(s+1)})\upharpoonright f(s+1)}(q) = 1$ with the computation being a correct one from $A \oplus K$ by the definition of $g(s+1) < f(s+1)$. Thus we would take the least such q to be $p_{s+1} \in D_n$ for the desired contradiction. ■

We now give a couple of applications that will play a crucial role in our global analysis of definability in $\mathcal{D}$ and, in particular, of the jump operator [??]. The first is a jump inversion theorem that ??strengthens and (check original)?? generalizes Shoenfield's ??.

ompleteness **Theorem 8.3.10 ($\overline{\mathbf{GL}}_2$ jump inversion)** *If $A \in \overline{GL}_2$, $C \geq_T A \vee 0'$, and C is r.e. in A, then there is $B \leq_T A$ such that $B' \equiv_T C$.*

Proof. Let C_s be an enumeration of C recursive in A. We want a notion forcing recursive in A and a collection of dense sets D_n such that for any $\langle D_n \rangle$ generic G, $G' \equiv_T C$. This time, our notion of forcing has conditions $p \in 2^{<\omega}$. The definition of extension for $\mathcal{P}$ is a bit tricky. If $q \supseteq p$ and

$$\langle e, x \rangle \in [|p|, |q|) \Rightarrow [C_{|p|}(x) = q(\langle e, x \rangle) \text{ or } \exists n \leq e\, (\Phi_n^p(n) \uparrow \ \& \ \Phi_n^q(n) \downarrow)]$$

we say that $q \leq_1 p$. Now this relation is clearly recursive in A since A computes $C_{|p|}$ for each p. However, it need not be transitive (Exercise ntran) ??. We let $\leq_{\mathcal{P}}$ be its transitive closure. As, given any $r \supseteq p$, there are only finitely many q's with $r \supseteq q \supseteq p$ we can check all possible routes via $\leq_1$ from p to r recursively in A and so $\leq_{\mathcal{P}}$ is also recursive in A. The plan for coding C into G' uses the Shoenfield limit lemma and partially explains the notion of extension. It guarantees that $e \in C \Rightarrow G^{[e]} =^* \omega$ while $e \notin C \Rightarrow G^{[e]} =^* \emptyset$. Thus $e \in C \Leftrightarrow \lim_s G(\langle e, s \rangle = 1$ and so $C \leq_T G'$. Suppose we have a generic sequence $\langle p_s \rangle \leq_T A$ for some collection of dense sets as in Theorem gl2genseq 8.3.9. The definition of extension guarantees that coding mistakes can happen in column e only when $\Phi_n^{p_s}(n)$ first converges for some $n \leq e$. Thus we will have $C \leq_T G'$.

Our first class of dense sets include the trivial requirements and in addition force the jump of G in the hope of making $G' \leq_T C$:

$$\begin{aligned} D_{m,j} &= \{p : |p| \geq j \ \& \ [\Phi_m^p(m) \downarrow \text{ or } (\forall q \supseteq p)(\Phi_m^q(m) \uparrow \\ \text{or } [(\exists e &< m)(\exists \langle e, x \rangle \in [|p|, |q|)(C_{|p|}(e) \neq q(\langle e, x \rangle) \text{ but } \neg(\exists n \leq e)(\Phi_n^p(n) \uparrow \ \& \ \Phi_n^q(n) \downarrow)])\} \end{aligned}$$

Note that after we use A to compute $C_{|p|}$, membership in $D_{m,j}$ is a Π^0_1 property and so recursive in $0'$. Thus, the $D_{m,j}$ are uniformly recursive in $A \vee 0'$. We must argue that they are dense. Consider any p. We can clearly extend it to a q with $|q| \geq j$ by making $q(\langle e,x\rangle) = C_{|p|}(e)$ for $\langle e,x\rangle \in [|p|, j)$. So we may as well assume that $|p| \geq j$. If $\Phi^p_m(m)\downarrow$ then $p \in D_{m,j}$ and we are done. So suppose $\Phi^p_m(m)\uparrow$. If there is $q \supseteq p$ such that $\Phi^q_m(m)\downarrow$ and $(\forall e < m)(\forall \langle e,x\rangle \in [|p|,|q|)[C_{|p|}(x) = q(\langle e,x\rangle)$ or $\exists n \leq e\,(\Phi^p_n(n)\uparrow\ \&\ \Phi^q_n(n)\downarrow)]$, $q \leq_{\mathcal{P}} p$ by definition. (because $\Phi^p_m(m)\uparrow$ while $\Phi^q_m(m)\downarrow$ so any violation of coding is allowed for $e \geq m$) and is in $D_{m,j}$. If there is no such q then $p \in D_{m,j}$ by definition.

Now we verify that $G = \cup p_s$ has the desired properties. By Theorem gl2genseq 8.3.9, $G \leq_T A$. To see that $C \leq_T G'$ consider any e. Let s be such that $(\forall i \leq e)(\Phi^G_i(i)\downarrow \Rightarrow \Phi^{p_s}_i(i)\downarrow$ & $i \in C \Rightarrow i \in C_{|p_s|})$. It is clear from the definition of $\leq_{\mathcal{P}}$ that for any $t > s$ and $\langle i,x\rangle \in [|p_s|,|p_t|)$ with $i \leq e$, $\langle i,x\rangle \in p_t \Leftrightarrow i \in C$. Thus $C(e) = \lim_t G(\langle e,t\rangle$ and so $C \leq_T G'$ by the Shoenfield limit lemma. For the other direction we want to compute $G'(e)$ recursively in C. (Of course, $A \leq_T C$ and so then is $\langle p_s\rangle$.) Suppose we have, by induction, computed an s as above for $e-1$. We can now ask if $e \in C$. If so. we find a $u \geq t \geq s$ such that $e \in C_{|p_t|}$ and $p_u \in D_{e,|p_t|}$. If $\Phi^{p_u}_e(e)\downarrow$, then, of course, $e \in G'$. If $\Phi^{p_u}_e(e)\uparrow$ but $e \in G'$, then there would be a $v > u$ such that $\Phi^{p_v}_e(e)\downarrow$ and, of course, $p_v \leq_{\mathcal{P}} p_u$. This would contradict the fact that $p_u \in D_{e,|p_t|}$ by our choice of s and t and the definitions of $D_{e,|p_t|}$ and $\leq_{\mathcal{P}}$. ■

Corollary 8.3.11 (Shoenfield Jump Inversion Theorem) *For all $C \geq 0'$ there is $B < 0'$ such that $B' \equiv_T C$ if and only if C is r.e. in $0'$.*

Proof. The "only if" direction is immediate. The "if" direction follows directly from the Theorem by taking $A = 0'$. ■

For later applications we now strengthen the above jump inversion theorem to make $B <_T A$.

ompleteness **Theorem 8.3.12** *If $A \in \overline{GL}_2$, $C \geq_T A \vee 0'$, and C is r.e. in A, then there is $B <_T A$ such that $B' \equiv_T C$.*

Proof. In addition to the requirements of Theorem gl2completeness 8.3.10, we need to make sure that $\Phi^G_i \neq A$ for each i. To do this we modify the definition of extension to also allow violations of the coding requirements for e when we newly satisfy one of these diagonalization requirements for $i \leq e$. (As we did above for making $\Phi^G_i(i)\downarrow$.) We say $q \leq_1 p$ if $\langle e,x\rangle \in [|p|,|q|) \Rightarrow [C_{|p|}(x) = q(\langle e,x\rangle)$ or $\exists n \leq e\,([\Phi^p_n(n)\uparrow\ \&\ \Phi^q_n(n)\downarrow]$ or $[\exists y \Phi^q_n(y)\downarrow \neq A(y)\ \&\ \neg\exists y \Phi^p_n(y)\downarrow \neq A(y)]$ Again $\leq_{\mathcal{P}}$ is defined as the transitive closure of this relation and it is recursive in $A \vee 0'$ as before. We then adjust the $D_{m,j}$ accordingly

$$\begin{aligned} D_{m,j} &= \{p : |p| > j\ \&\ [\Phi^p_m(m)\downarrow \text{ or } (\forall q \supseteq p)(\Phi^q_m(m)\uparrow \\ \text{or } [(\exists e &< m)(\exists \langle e,x\rangle \in [|p|,|q|)(C_{|p|}(e) \neq q(\langle e,x\rangle) \text{ but} \\ \neg(\exists n &\leq e)([\Phi^p_n(n)\uparrow\ \&\ \Phi^q_n(n)\downarrow]\ \&\ \neg(\exists y)[\Phi^q_n(y)\downarrow \neq A(y)\ \&\ \neg\exists y \Phi^p_n(y)\downarrow \neq A(y)])]]\}. \end{aligned}$$

We also need dense sets that guarantee that $\Phi_e^G \neq A$:

$$\begin{array}{rcl} D_i & = & \{p|(\exists x)(\Phi_i^p(x)\downarrow\neq A(x) \text{ or} \\ (\forall q_0, q_1 & \supseteq & p)(\forall x < |q_0|, |q_1|)[\neg(\Phi_i^{q_0}(x)\downarrow\neq \Phi_i^{q_1}(x)\downarrow) \text{ or} \\ ((\exists e & < & i)(\exists\langle e, x\rangle \in [|p|, |q|)(\exists j \in \{0,1\})[(C_{|p|}(e) \neq q_i(\langle e, x\rangle) \text{ but} \\ \neg(\exists n & \leq & i)([\Phi_n^p(n)\uparrow \ \& \ \Phi_n^q(n)\downarrow] \ \& \ \neg(\exists y)[\Phi_n^q(y)\downarrow\neq A(y) \ \& \ \neg\exists y \Phi_n^p(y)\downarrow\neq A(y)])]\}. \end{array}$$

The proof now proceeds as in the previous Theorem. The arguments for all the verifications are now essentially the same as there and are left as an exercise.?? ■

Exercise 8.3.13 *Verify that the notion of forcing and classes of dense sets specified in proof of Theorem* stgl2completeness *8.3.12 suffice to actually prove it.*

Exercise 8.3.14 *Prove that if A is r.e. and $C \geq_T 0'$ is r.e. in A then there is a $B \leq_T A$ such that $B' \equiv_T C$. Indeed we may also make $B <_T A$. Hint:*

The next result says that every $\mathbf{a} \in \overline{\mathbf{GL}}_2$ is **RRE** (*relatively recursively enumerable*), i.e. there is a $\mathbf{b} < \mathbf{a}$ such that $\mathbf{a}$ is r.e. in $\mathbf{b}$ and a bit more.

gl2rre **Theorem 8.3.15** *If $\mathbf{a} \in \overline{\mathbf{GL}}_2$ then there is $\mathbf{b} < \mathbf{a}$ such that $\mathbf{a}$ is r.e. in $\mathbf{b}$ and $\mathbf{a}$ is in $\overline{\mathbf{GL}}_2(\mathbf{b})$, i.e. $(\mathbf{a} \vee \mathbf{b}')' < \mathbf{a}''$.*

Proof. Let $\mathbf{a} \in \overline{\mathbf{GL}}_2$. We'll use a notion of forcing $\mathcal{P}$ with conditions $p = \langle p_0, p_1, p_2\rangle$, $p_i \in 2^{<\omega}$ such that

1. $|p_0| = |p_1|$, $p_0(d_n) = A(n)$, $p_1(d_n) = 1 - A(n)$ where d_n is nth place where p_0, p_1 differ and
2. $(\forall e < |p_0 + p_1|)(e \in p_0 \oplus p_1 \Leftrightarrow \exists x(\langle e, x\rangle \in p_2))$

As expected, our generic set $G_0 \oplus G_1 \oplus G_2$ is given by $V(p) = p_0 \oplus p_1 \oplus p_2$. The idea here is that if we can force p_0, p_1 to differ at infinitely many places while still making our generic sequence recursive in A, the first clause in the definition of $\leq_{\mathcal{P}}$ guarantees that $G_0 \oplus G_1 \equiv_T A$. The second clause works towards making $G_0 \oplus G_1$ r.e. in G_2 with the intention being that $\deg(G_2) = \mathbf{g}_2$ is to be the $\mathbf{b}$ required by the theorem. Extension in the notion of forcing is defined in the simplest way as $q \leq_{\mathcal{P}} p \Leftrightarrow q_i \supseteq p_i$ but note that this only applies to p and q in $\mathcal{P}$ and not all q with $q_i \supseteq p_i$ are in $\mathcal{P}$ even if $p \in \mathcal{P}$. The notion of forcing is clearly recursive in A.

We now define the dense sets needed to satisfy the requirements of the Theorem. We begin with $D_{2n} = \{p : p_0, p_1 \text{ differ at at least } n \text{ points}\}$. These sets are clearly recursive in A. We argue that these are dense by induction on n. Suppose D_{2n} is dense. To show that D_{2n+2} is dense, it suffices, for any given $p \in D_{2n} - D_{2n+2}$ to find a $q \leq_{\mathcal{P}} p$ in D_{2n+2}. Let $q_0 = p_0\hat{\ }A(n)$, $q_1 = p_1\hat{\ }(1 - A(n))$. Choose $i \in \{0, 1\}$ such that $q_i(|p_0|) = 1$.

Define $q_2 \supseteq p_2$ by choosing x large and setting $q_2(\langle 2|p_0| + i, x\rangle) = 1$ and $q_2(z) = 0$ for all $z \notin \mathrm{dom}(p_2)$ and less than $\langle 2|p_0| + i, x\rangle$. Now $q = \langle q_0, q_1, q_2\rangle$ satisfies the requirements to be a condition in P. It obviously extends p and is in D_{2n+2}.

For any generic recursive in A which meets all the D_{2n}, $G_0 \oplus G_1 \equiv_T A$ and $G_0 \oplus G_1$ is r.e. in G_2.

We also want dense sets similar in flavor to those of the previous theorems to force the jump of G_2 to make $(\mathbf{a} \vee \mathbf{g}_2')' < \mathbf{a}''$. Let

$$\begin{aligned} D_{2n+1} &= \{p : \Phi_n^{p_2}(n) \downarrow \text{ or } (\forall \sigma \supseteq p_2) \\ &\quad (\Phi_n^{\sigma}(n) \uparrow \text{ or } (\exists \langle e, x\rangle \in \sigma)((p_0 \oplus p_1)(e) = 0)\,. \end{aligned}$$

For $p \in P$, membership in D_{2n+1} is a $0'$ question and so these sets are recursive in $A \vee 0'$. We want to prove that they are dense. Suppose have a $p \in P$ so we want a $q \leq_{\mathcal{P}} p$ with $q \in D_{2n+1}$. We may suppose that $\Phi_n^{p_2}(n) \uparrow$ and that the second clause fails for p as otherwise we would already be done. Thus we have a $\sigma \supseteq p_2$ such that $\Phi_n^{\sigma}(n) \downarrow$ but $\neg(\exists \langle e, x\rangle \in \sigma)((p_0 \oplus p_1)(e) = 0)$. We claim that there is a $q \leq_{\mathcal{P}} p$ such that $q_2 \supseteq \sigma$ and so $\Phi_n^{q_2}(n) \downarrow$ and $q \in D_{2n+1}$ as required. The only issue is that there may be some $\langle j, y\rangle \in \sigma$ with $j > |p_0 \oplus p_1|$. If so we must define q_0 and q_1 accordingly, i.e. $j \in q_0 \oplus q_1$. So if j is even, we want $\frac{j}{2} \in q_0$; if it is odd, $\frac{j-1}{2} \in q_1$. We now define q_0, q_1 at the appropriate element ($\frac{j}{2}$ or $\frac{j-1}{2}$) to both be 1. Elsewhere we let both q_0 and q_1 be 0. Thus we have not added any points at which q_0 and q_1 differ beyond those in p_0, p_1). Now we extend σ to q_2 by adding $\langle e, y\rangle$ for some large y if $(q_0 \oplus q_1)(e) = 1$ and $e \geq |p_0 \oplus p_1|$ and wherever not yet defined we let $q_2(z) = 0$. Thus $q \in P$ and is the desired extension of p in D_{2n+1} as $\Phi_n^{q_2}(n) = \Phi_n^{\sigma}(n) \downarrow$.

We now let $\langle p_s\rangle \leq_T A$ be a generic sequence meeting every D_n as given by Theorem gl2genseq 8.3.9. We already have seen that $G_0 \oplus G_1 \equiv_T A$ and is r.e. in $G_2 \leq_T A$. If we can show that $(A \oplus G_2')' <_T A''$ then we will be done as this clearly implies that $G_2 <_T A$. We first claim that $G_2' \leq_T A \vee 0'$. To see if $n \in G_2'$, recursively in $A \vee 0'$ find an s such that $p_s \in D_{2n+1}$. Then we claim that $n \in G_2' \Leftrightarrow \Phi_n^{p_{s,2}}(n) \downarrow$. If $\Phi_n^{p_2}(n) \downarrow$, then we are done. If not, then $(\forall \sigma \supseteq p_{s,2})\,(\Phi_n^{\sigma}(n) \uparrow$ or $(\exists \langle e, x\rangle \in \sigma)((p_0 \oplus p_1)(e) = 0))$ and by definition of membership and extension in $\mathcal{P}$, $\Phi_n^{p_{t,2}}(n) \uparrow$ for every $p_{t,2}$ for $t \geq s$. Thus $\Phi_n^{G_2}(n) \uparrow$ as desired. As $G_2' \leq_T A \vee 0'$, $(A \oplus G_2') = A \vee 0'$ and so as $A \notin GL_2$, $(A \oplus G_2')' = (A \vee 0')' <_T A''$ as required. ■

Exercise 8.3.16 *If $A >_T 0$ is r.e. and $C \geq_T 0'$ is r.e. in A then there is a $B \leq_T A$ such that $B' \equiv_T C$. Indeed we may also make $B <_T A$. Hint: ....build β_s finite extension obey coding rule for columns for $e \leq c(s) \leq s$ (enumerates C recursively in A) except that can violate to force jump as above; search below $m_A(s+1)$ for extensions forcing jump for $e \leq s$ that obey rule. Also search for extensions so Φ_e giving different answers and allow violations in columns $> e$ when satisfy this requirement by choosing one that gives answer other than A ??*

We can now deduce a result that will play a major role in our definition of the Turing jump in $\mathcal{D}$ and many related results.

Theorem 8.3.17 *If $A \in \overline{GL}_2$ and $S \in \Sigma_3^A$ then there is an embedding of an effective successor model (with the appropriate partial lattice structure) in the degrees below* $\deg(A)$ *and an exact pair $\mathbf{x}, \mathbf{y}$ for the ideal generated by the $\mathbf{d}_n$ with $n \in S$. (Remember that the $\mathbf{d}_n$ are the degrees representing $n \in \mathbb{N}$ in the effective successor model.*

Proof. Given $A \in \overline{GL}_2$ and $S \in \Sigma_3^A$ Theorem gl2rre 8.3.15 gives us a $B < A$ such that A is r.e. in B and A is $\overline{GL}_2(B)$. Since $A' \geq A \vee 0'$ and is r.e. in it, Theorem gl2completeness 8.3.10 relativized to B gives us a $\hat{B} < A$ (with $B \leq_T \hat{B}$) such that $B' \equiv A'$ and so $\Sigma_3^{\hat{B}} = \Sigma_3^A$, Moreover, A is r.e. in $\hat{B}$ because it was r.e. in $B \leq_T \hat{B}$. The result now follows from Theorem ?? to embed an effective successor model between $\hat{B}$ and A and Theorems resigma3ideal 8.2.12 to pick out the ideal generated by the associated $\mathbf{d}_n$ for $n \in S$ as the set $\{e | \exists n (\Phi_e^{\hat{B}} \in \mathbf{d}_n)\}$ is itself $\Sigma_3^{\hat{B}} = \Sigma_3^A$ as is then $\{e | (\exists n \in S)(\Phi_e^A \in \mathbf{d}_n)\}$. ■

Below a $\mathbf{H}_1$ or $\mathbf{GH}_1$ degree?? Minimal degree in ?? others here??complementation??

Exercise 8.3.18 *Prove that every degree has a $\mathbf{GL}_2$ degree below it.*

Exercise 8.3.19 *Prove that every degree has a $\mathbf{GL}_1$ degree above it.*

Exercise 8.3.20 *Prove that every recursive lattice $\mathcal{L}$ with 0 and 1 can be embedded in $\mathcal{D}(\leq \mathbf{a})$ preserving 0 and 1 for any $\mathbf{a} \in \overline{\mathbf{GL_2}}$.*

Other results of this type? Lerman, Antonio?

History: most in Jockusch Posner 1978 at least for Cohen forcing.

8.4 Array Nonrecursive Degrees

The notion of array nonrecursiveness was originally introduced in the context of r.e. degrees to capture certain types of arguments in which one needed multiple permissions from (changes in) a given r.e. set to construct a desired set. (DJS I) It was phrased in terms of the r.e. set meeting (intersecting) the elements of certain types of arrays of uniformly given finite sets. It was later (DJS II) generalized to all degrees with a definition based on a domination property involving functions weak truth-table reducible to $0'$ and shown to have many of the properties of $\overline{\mathbf{GL}}_2$ degrees.

anrdeg **Definition 8.4.1** *A degree* $\mathbf{a}$ *is* **ANR** *if for every function $g \leq_{wtt} 0'$ there is an $f \leq_T \mathbf{a}$ such that f is not dominated by g.*

Exercise 8.4.2 *If* $\mathbf{a} \in \overline{\mathbf{GL_2}}$ *then* $\mathbf{a} \in \mathbf{ANR}$.

This notion is actually equivalent to two related ones, one seemingly weaker and the other seemingly stronger. (DJS and CSh) anreq **Proposition 8.4.3** *The following are equivalent for a degree* $\mathbf{a}$*:*

1. $\mathbf{a}$ is **ANR**.
2. There is a function $f \leq_T \mathbf{a}$ which is not dominated by the least modulus function m_K for $0'$.
3. For any $A \in \mathbf{a}$ and $g = \Phi_e(A \oplus 0')$ such that there is a function $r \leq_T A$ bounding the use from $0'$ in the computation of g at each x, there is a $k \leq_T A$ which is not dominated by g.

Proof. That (1) implies (2) and (3) implies (1) are immediate from the definitions. We prove that (2) implies (3).

Without loss of generality we may assume that f, g and r are increasing. We define the required $k \leq_T A$ as follows: To calculate $k(n)$ compute, for each $s > n$ in turn, $\Phi_{e,fr(s)}(A \oplus 0'_{fr(s)}; n)$ (i.e. compute $fr(s)$ many steps in the standard enumeration of $0'$ and then, using this set as the second component of the oracle (and A for the first), compute Φ_e at n for $fr(s)$ many steps) until the computation converges and then add 1 to get the value of $k(n)$. This procedure must converge as $\Phi_e(A \oplus 0'; n)$ converges. Now, as m_K does not dominate f, there are infinitely many n such that there is a $j \in [r(n), r(n+1))$ with $f(j) > m_K(j)$. For such n we have $fr(n+1) > f(j) > m_K(j) \geq m_K r(n)$. Thus $0'_{fr(s)} \restriction r(n) = 0' \restriction r(n)$ for every $s > n$. So the computation of $\Phi_e(A \oplus 0'; n)$ is, step by step, the same as that of $\Phi_e(A \oplus 0'_{fr(s)}; n)$ for each $s > n$ as all the oracles agree on the actual use of the true computation. So eventually we get an $s > n$ such that $\Phi_{e,fr(s)}(f \oplus 0'_{fr(s)}; n) \downarrow$ and the output must be $\Phi_e(A \oplus 0'; n)$. Thus, for these n, $k(n) = g(n) + 1 > g(n)$ as required. ■

lowanr **Exercise 8.4.4** *There is an* $\mathbf{a} \in$ **ANR** *with* $\mathbf{a} \in \mathbf{L}_1$. *In fact, there is a Cohen* 1*-generic* A *whose degree is* **ANR**. *Hint: use Proposition* anreq *8.4.3(2) and the principal function ??definition?? of* A.

Exercises on f is ANR, relativizations and uniformity

That there are Cohen 1-generics below every $\mathbf{a} \in$ **ANR** follows immediately from the proof of Theorem g121gen 8.3.3 and Remark cohenanr 8.3.4. This, as usual, gives one whole array of corollaries. We now prove the analog for **ANR** of the stronger version given for $\overline{\mathbf{GL}}_2$ degrees in Theorem gl2genseq 8.3.9. This allows us to carry out almost all of the known forcing constructions for $\overline{\mathbf{GL}}_2$ degrees for **ANR** ones.

anrgenseq **Theorem 8.4.5** *If* A *is of* **ANR** *degree,* $\mathcal{P}$ *is an* A*-recursive notion of forcing,* $\mathcal{C} = \langle D_n \rangle$ *a sequence of sets dense in* $\mathcal{P}$ *(including the ones* $\{p |\ |V(p)| > l\}$ *for each* l*) with a density function* $d(x,y) = \Psi(A \oplus 0'; x, y)$ *such that the use from* $0'$ *in the computation of* $\Psi(A \oplus 0'; x, y)$ *is bounded by a function* $\hat{r} \leq_T A$*, then there is a* $\mathcal{C}$*-generic sequence* $\langle p_s \rangle$ *recursive in* A*. Indeed,* $\forall n \exists s (p_{s+1} = d(p_s, n))$*.*

Proof. Without loss of generality we may assume that $\hat{r}(x, y)$ is increasing in both x and y. Next note that the nondecreasing function $m_K\hat{r}(s, s)$ satisfies the hypotheses of Proposition $\overset{\texttt{anreq}}{8.4.3}$(3), i.e. it is computable from $A \oplus 0'$ and its $0'$ use is bounded by a function $(\hat{r}(s, s))$ recursive in A. Finally note that the maximum of the running times of $\Psi(A \oplus 0'; x, y)$ for $x, y \leq s$ is also is such a function. (We run Ψ on each input and then output the sum of the number of steps needed to converge.) Finally, we let r be the maximum of these three functions so it too is of the desired form. By Proposition $\overset{\texttt{anreq}}{8.4.3}$, we now have an increasing function $g \leq_T A$ not dominated by r. We use g to construct the desired generic sequence p_s by recursion.

We begin with $p_1 = \mathbf{1}$. At step $s + 1$ we have (by induction) a nested sequence $\langle p_i | i \leq s\rangle$ with $p_i \leq s$. We calculate $0'_{g(s+1)}$ and see if there are any changes on the use from $0'$ in a computation based on which some D_m was previous declared satisfied. If so, we now declare it unsatisfied. Suppose n is the least $m < s + 1$ such that D_m is not now declared satisfied. (There must be one as we declare at most one m to be satisfied at every stage and none at stage 1.) We compute $\Psi_{g(s+1)}(A \oplus 0'_{g(s+1)}; p_s, n)$. If the computation does not converge or gives an output q such that $q > s + 1$ or $q \not\leq_{\mathcal{P}} p_s$ we end the stage and set $p_{s+1} = p_s$. Otherwise, we end the stage, declare D_n to be satisfied on the basis of this computation of the output q and set $p_{s+1} = q$. Of course, $\langle p_s\rangle \leq_T A$.

We now verify that$\langle p_s\rangle$ is $\mathcal{C}$-generic and indeed $\forall n \exists s (p_{s+1} = d(p_s, n))$. Clearly if we ever declare D_n to be satisfied (and define p_{s+1} accordingly) and it never becomes unsatisfied again then $p_{s+1} = d(p_s, n)$. Moreover, if we ever declare D_n to be satisfied (and define p_{s+1} accordingly) and it remains satisfied at a point of the construction at which we have enumerated $0'$ correctly up to $r(p_s, n)$, then by definition $p_{s+1} = d(p_s, n)$ and D_n is never declared unsatisfied again. We now show that this happens.

Suppose all D_m for $m < n$ have been declared satisfied by s_0 and are never declared unsatisfied again. Let $s+1 \geq s_0$ be least such that $g(s+1) \geq r(s+1)$. If D_n was declared satisfied at some $t + 1 \leq s$ on the basis of some computation of $\Psi_{g(t+1)}(A \oplus 0'_{g(t+1)}; p_t, n)$ and there is no change in $0'$ on the use of this computation by stage $g(s + 1)$ then the computation is correct, $p_{t+1} = \Psi(A \oplus 0'; p_t, n) \in D_n$ and D_n is never declared unsatisfied again. (The point here is that by our choice of s, $g(s+1) > m_K r(s+1, s+1) \geq m_K r(p_t, n)$ and so $0'_{g(s)} \upharpoonright r(p_t, n) = 0' \upharpoonright r(p_t, n)$.) Otherwise, D_n is unsatisfied at s and the least such. By construction we compute $\Psi_{g(s+1)}(A \oplus 0'_{g(s+1)}; p_s, n)$. The definition of r along with our choice of g and s guarantee that this computation converges and is correct and so unless $d(p_s, n) > s + 1$ we declare D_n satisfied, set $p_{s+1} = d(p_s, n)$ and D_n is never declared unsatisfied again. If $d(p_s, n) > s + 1$, we set $p_{s+1} = p_s$ and, as D_n remains unsatisfied and the computations already found do not change, we continue to do this until we reach a stage $v + 1 \geq d(p_s, n)$ at which point $p_v = p_s$ and we set $p_{v+1} = d(p_v, n)$ declare D_n satisfied and it is never unsatisfied again. ■

Exercise 8.4.6 *Prove that every recursive lattice $\mathcal{L}$ with* 0 *and* 1 *can be embedded in $\mathcal{D}(\leq \mathbf{a})$ preserving* 0 *and* 1 *for any $\mathbf{a} \in \mathbf{ANR}$. (DJS)*

Exercise 8.4.7 *Prove that every $\mathbf{a} \in \mathbf{ANR}$ has the cupping property.*

jump inversion others Exercises??

Our goal now is to characterize the **ANR** degrees as those degrees **a** such that every $\mathbf{b} \geq \mathbf{a}$ is **RRE**. We begin with the analog of Theorem g12rre 8.3.15 which provides one half of the equivalence.

anrrre **Theorem 8.4.8** *If* $\mathbf{a} \in \mathbf{ANR}$ *then* **a** *is* **RRE**.

Proof. We use an A-recursive notion of forcing $\mathcal{P}$ with conditions $p = \langle p_0, p_1, p_2 \rangle$, $p_i \in 2^{<\omega}$ such that

1. $|p_0| = |p_1|$, $p_0(d_n) = A(n-1)$, $p_1(d_n) = 1 - A(n-1)$ where d_n is n^{th} place where p_0, p_1 differ and
2. $(\forall e < |p_0 \oplus p_1|)(e \in p_0 \oplus p_1 \Leftrightarrow \exists x(\langle e, x \rangle \in p_2))$.

Extension in this notion of forcing is defined simply by $q \leq_{\mathcal{P}} p \Leftrightarrow q_i \supseteq p_i$ but note that this applies only to p and q in P. Membership in P and $\leq_{\mathcal{P}}$ are clearly recursive in A.

Our plan is to define a class $\mathcal{C}$ of dense sets D_n with a density function $d(p, n)$ recursive in $A \oplus 0'$ with $0'$ use recursively bounded. Theorem anrgenseq 8.4.5 then supplies a $\mathcal{C}$-generic sequence $\langle p_s \rangle \leq_T A$ from which we can define the required $G \leq_T A$ in which **a** is r.e. If $p_s = \langle p_{s,0}, p_{s,1}, p_{s,2} \rangle$ we let $G_i = \cup\{p_{s,i} | s \in \mathbb{N}\}$ for $i = 0, 1, 2$ so $G_i \leq_T A$. Then, if we can force G_0 and G_1 to differ at infinitely many places, $G_0 \oplus G_1 \equiv_T A$. On the other hand, the definition of the notion of forcing obviously makes $G_0 \oplus G_1$ r.e. in G_2. Thus **a** will be r.e. in $\mathbf{g} = \deg(G_2)$. We will have other requirements that make $\mathbf{g} < \mathbf{a}$ as well.

We begin with the dense sets that provide the differences we need:

$$D_{2n} = \{p \in \mathcal{P} : p_0, p_1 \text{ differ at at least } n \text{ points}\}.$$

We define the required function $d(r, 2n)$ by recursion on n. Given r and $n+1$, we suppose we have calculated $d(r, 2n) = p = \langle p_0, p_1, p_2 \rangle \in D_{2n}$ with $p \leq_{\mathcal{P}} r$. If $p \notin D_{2n+2}$, we need to compute a $q = \langle q_0, q_1, q_2 \rangle \in D_{2n+2}$ with $q \leq_{\mathcal{P}} p$. Let $q_0 = p_0\hat{\ }A(n)$, $q_1 = p_1\hat{\ }(1 - A(n))$. Choose $i \in \{0, 1\}$ such that $q_i(|p_0|) = 1$. Define $q_2 \supseteq p_2$ by choosing x large and setting $q_2(\langle 2|p_0| + i, x \rangle) = 1$ and $q_2(z) = 0$ for all $z \notin \operatorname{dom}(p_2)$ and less than $\langle 2|p_0| + i, x \rangle$. Now $q = \langle q_0, q_1, q_2 \rangle$ satisfies the requirements to be a condition in P. It obviously extends p and is in D_{2n+2}. This computation is clearly recursive in A.

We must now add dense sets to guarantee that $A \not\leq_T G_2$:

$$\begin{aligned} D_{2n+1} &= \{p \in \mathcal{P} :\ \exists x(\Phi_n^{p_2}(x) \downarrow \neq A(x)) \text{ or } \forall(\sigma_0, \sigma_1 \supseteq p_2)[\exists x(\Phi_n^{\sigma_0}(x) \downarrow \neq \Phi_n^{\sigma_1}(x)) \downarrow \Rightarrow \\ (\exists i &\in \{0,1\})(\exists \langle e, x \rangle)(e < |p_0 \oplus p_1| \;\&\; \sigma_i(\langle e, x \rangle) = 1 \neq (p_0 \oplus p_1)(e)]\}. \end{aligned}$$

Of course, the first alternative guarantees that $\Phi_n^{G_2} \neq A$ while the second that $\Phi_n^{G_2}$, if total, is recursive. The point here is that if some p_s in our generic sequence satisfies the second clause then, we can, for any z, calculate $\Phi_n^{G_2}(z)$ by finding any $\sigma \supseteq p_{s,2}$ such that $\Phi_n^{\sigma}(z) \downarrow$ and taking its value as $\Phi_n^{G_2}(z)$. There is such a $\sigma \subseteq G_2$ as $\Phi_n^{G_2}$ is assumed to be total and $G_2 \supseteq p_{s,2}$. If there were some other $\tau \supseteq p_{s,2}$ with $\Phi_n^\tau(z) \downarrow \neq \Phi_n^\sigma(z) \downarrow$ then, by our choice of s and the definition of D_{2n+1}, there is no $\langle e, x\rangle$ with $e < |p_0 \oplus p_1|$ such that $\tau(\langle e, x\rangle) = 1 \neq (p_0 \oplus p_1)(e)$. Thus we could form a condition $q \leq_{\mathcal{P}} p_s$ with $q_2 = \tau$ by extending p_0 and p_1 by setting $q_1(w) = q_2(w) = 1$ (for $w \geq |p_0|$) if either $\langle 2w, v\rangle$ or $\langle 2w+1, v\rangle$ is in τ for any v. In this way no new differences between q_0 and q_1 (not already in p_0 and p_1) occur and the definition of being a condition is satisfied. Thus q is a condition extending $p_{s,2}$ with $\Phi_n^{q_2}(z) \downarrow \neq A(z)$ contradicting our choice of s.

We compute the required density function $d(q, 2n+1)$ as follows. Given q we ask one question of $0'$ determined recursively in q: Are there extensions σ_0, σ_1 of q_2 that would show that q does not satisfy the second disjunct in the definition of D_{2n+1}. If not, let $d(q, 2n+1) = q$ which is already in D_{2n+1}. If so, we find the first such pair (appearing in a recursive search) and ask A which σ_i gives an answer different from $A(x)$. We now need a condition $r = d(q, 2n+1)$ extending q with third coordinate r_2 extending σ_i. For each $\langle e, x\rangle$ with $e \geq |q_1 \oplus q_2|$) and $\sigma_i(\langle e, x\rangle) = 1$ we define $r_j(z) = 1$ for both $j \in \{0, 1\}$ for the z that makes $(r_0 \oplus r_1)(e) = 1$ and otherwise we let $r_j(u) = 0$ for all other u less than the largest element put into either r_0 or r_1 by the previous procedure. We now extend σ_i to the desired r_2 by putting in $\langle k, y\rangle$ for a large y for all those $k \geq |q_1|$ put into $r_0 \oplus r_1$ for which there is no $\langle k, w\rangle$ in σ_i. Otherwise we extend σ_i by 0 up to the largest element put in by this procedure. It is clear that this produces a condition r as required. (No points of difference between r_0 and r_1 are created that were not already present in q.)

We now apply Theorem anrgenseq 8.4.5 to get a $\mathcal{C}$-generic sequence $\langle p_s\rangle \leq_T A$. As promised, we let $G_i = \cup\{p_{s,i} | s \in \mathbb{N}\}$ for $i = 0, 1, 2$ and, as described above, $A \equiv_T G_0 \oplus G_1$ which is r.e. in G_2. In addition, the conditions in D_{2n+1} guarantee (as above) that $\Phi_n^{G_2} \neq A$ as well. ■

Exercise 8.4.9 *Prove that every* $\mathbf{a} \in \mathbf{ANR}$ *has the cupping property. Hint? Indifference set, i.e.* $f : \mathbb{N} \to \{0, 1, 2\}$ *approach??*

characterization as all above are RRE reference notions and terminology about trees from §spectormin 9.2

??Exercises on Relativization via Proposition mod ??:

anrreldef **Definition 8.4.10** *A function f is ANR if it is not dominated by m. It is ANR relative to h if $h \leq_T f$ and f is not dominated by m_h. A degree* $\mathbf{a}$ *is* **ANR** *relative to* $\mathbf{b}$, $\mathbf{ANR}(\mathbf{b})$, *if there are $f \in$* $\mathbf{a}$ *and $h \in$* $\mathbf{b}$ *such that f is ANR relative to h, ANR(h).*

Chapter 9

Minimal Degrees and Their Jumps

9.1 Introduction

We now return to extension of embeddings problem. We saw that as long as we do not attempt to put a new degree in the extension below a given degree, then anything consistent is possible (??Exercise extemb 5.2.10). We now turn toward the issue of whether one can put new degrees below given ones. The answer is strongly negative. In fact, strong enough so that we can rule out all the extensions not constructed by ??Exercise extemb 5.2.10 for finite lattices $\mathcal{P}$. Clearly embedding every finite lattice $\mathcal{P}$ as an initial segment of $\mathcal{D}$ suffices as then if $\mathcal{Q}$ adds elements below any of $\mathcal{P}$ then there can be no extension to $\mathcal{Q}$ of the embedding of $\mathcal{P}$ as an initial segment. We prove this and more in Chapter initialseg 10. This will suffice to decide the truth of all two quantifier sentences in $\mathcal{D}$ (Chapter 2qtth 10.4) and also to show that the set of true three quantifier sentences is not decidable (Chapter 3qtth 10.5.

We begin with the simplest case.

Definition 9.1.1 *A degree* $\mathbf{a} > \mathbf{0}$ *is minimal if, for any* $\mathbf{b} \leq \mathbf{a}$, $\mathbf{b} = \mathbf{0}$ *or* $\mathbf{b} = \mathbf{a}$. *A degree is* $\mathbf{a}$ *is a minimal cover of* $\mathbf{c} > \mathbf{a}$ *if for any* $\mathbf{b}$ *with* $\mathbf{c} \leq \mathbf{b} \leq \mathbf{a}$, $\mathbf{b} = \mathbf{c}$ *or* $\mathbf{b} = \mathbf{a}$.

We cannot hope to construct a set of minimal degree by forcing with finite conditions like Cohen forcing as we have seen that generics for such forcings have every countable partial order embedded below them. We move then from approximations (conditions) that are clopen sets in Cantor space (all extensions of a $\sigma \in 2^{\omega}$) to ones that are prefect subsets instead.

9.2 Perfect forcing and Spector minimal degrees

spectormin

We represent perfect subsets of Cantor space, $2^{\mathbb{N}}$ (i.e. nonempty sets with every point a limit point) by binary perfect (i.e. always branching) trees T (with no dead ends). The perfect subsets of Cantor space are then the paths $[T]$ through these trees. We present such trees as functions $T : 2^{<\omega} \to 2^{<\omega}$ with certain properties. ?? define Cantor space and relevant topology perfect trees etc. early on??

binfunctree **Definition 9.2.1** *A* binary function tree *is a (possibly partial) function $T : 2^{<\omega} \to 2^{<\omega}$ such that*

1. *$\sigma \subseteq \tau \Rightarrow T(\sigma) \subseteq T(\tau)$ (for $\tau \in \operatorname{dom}(T)$, so, in particular, if $T(\tau)\downarrow$ and $\sigma \subseteq \tau$ then $T(\sigma)\downarrow$) and*
2. *$\sigma|\tau \Rightarrow T(\sigma)|T(\tau)$ (for $\sigma, \tau \in \operatorname{dom}(T)$).*

Definition 9.2.2 *We say that a binary string τ is on T if there is a σ such that $T(\sigma) = \tau$. We say that τ is on T above ρ if there is a $\sigma \supseteq \rho$ with $T(\sigma) = \tau$.*

lth **Exercise 9.2.3** *If T is a binary function tree then (for $\sigma \in \operatorname{dom} T$), $|T(\sigma)| \geq |\sigma|$.*

Exercise 9.2.4 *If T is a binary tree in the sense of Definition* $\overset{\text{tree}}{4.2.1}$ *then $[T]$ is perfect if and only if there is a binary function tree S such that $[S] = [T]$. If T is recursive (as a subset of $2^{<\omega}$) then we may take S to be so as well.*

Definition 9.2.5 *We define an order $\leq_{\mathcal{S}}$ on the binary function trees by $S \leq_{\mathcal{S}} T \Leftrightarrow \forall\sigma(S(\sigma)\downarrow \Rightarrow \exists\tau(S(\sigma) = T(\tau))$, in which case we say that S is a* subtree *of T. In this chapter all trees will be partial recursive binary function trees (unless otherwise specified) and we will just call them trees. In this section they will also be total unless otherwise specified.*

Exercise 9.2.6 *If S and T are trees then $[S] \subseteq [T]$ if and only if $\forall\sigma\exists\tau(S(\sigma) \subseteq T(\tau)$.*

Our forcing conditions, in this section, will be these trees. The order relation $S \leq_{\mathcal{S}} T$ is then equivalent to $\forall\sigma\exists\tau(S(\sigma) = T(\tau))$.

The function V required in the definition of a notion of forcing is given by $V(T) = T(\emptyset)$ but the notion of extension makes it clear that the only possible generic sets G extending the condition T are the $G \in [T]$. This notion $\mathcal{S}$ of forcing with perfect recursive binary function trees is often called *Spector forcing*. Its analog in set theory is often called Sacks forcing or perfect forcing. Note that this notion of forcing is only recursive in $0''$. The crucial point here is that it takes $0''$ to determine if Φ_e is total. Once we know it is total, $0'$ suffices to determine if it is a binary function tree as this is then a Π^0_1 property. If S, T are conditions in $\mathcal{S}$ then $0'$ can also determine if $S \leq_{\mathcal{S}} T$ as this too is a Π^0_1 property. The point here is that if there is any τ such that $T(\tau) = S(\sigma)$ then it must be of length at most $|S(\sigma)|$ by Exercise $\overset{\text{lth}}{9.2.3}$

The requirements for a set G to be of minimal degree are as follows:

- N_e: $G \neq \Phi_e$ and

- M_e: If Φ_e^G is total then either Φ_e^G is recursive or $G \leq_T \Phi_e^G$.

The N_e requirements are very easy to meet.

spdiag **Lemma 9.2.7** *For each e the set of conditions $\{T | T \Vdash \neg(\Phi_e = G)\}$ is dense in $\mathcal{S}$. In fact the smaller set $D_e = \{T | \neg(\Phi_e = T(\emptyset)(x))\}$ is already dense in $\mathcal{S}$.*

Proof. Given any tree T and Φ_e, note that $\neg(T(i) = \Phi_e)$ for i at least one of 0 or 1 as $T(0)|T(1)$. Thus we may take as the desired extension S of T the subtree such that $S(\sigma) = T(i\hat{}\sigma)$, i.e. it starts with $T(i)$ for the appropriate i and then continues on as does T. ■

We formalize the operation that provides a witness to the density required in Lemma spdiag 9.2.7:

Definition 9.2.8 *For any partial tree T and $\sigma \in 2^{<\omega}$, the* full subtree *of T above σ, $Fu(T,\sigma)$ or sometimes simply T_σ, is the tree S defined by $S(\tau) = T(\sigma\hat{}\tau)$.*

Proposition 9.2.9 *If T is (partial) recursive then so is T_σ and an index for it can be found uniformly recursively in one for T.*

Proof. Immediate. ■

Proposition 9.2.10 *There are density functions for the D_e of Lemma spdiag 9.2.7 which are uniformly recursive in $0'$ on the set of (recursive binary function) trees.*

Proof. Given any T, find an x such that $T(\sigma\hat{}0)(x) \neq T(\sigma\hat{}1)(x)$. Then ask $0'$ if $\Phi_e(x)\downarrow$. If so compute its value. In any case take $i \in \{0,1\}$ such that $\neg(T(\sigma\hat{}i)(x) = \Phi_e(x))$ and take $Fu(T,\sigma\hat{}i)$ as the desired extension. ■

We must now see how to satisfy the minimality requirements M_e. We have seen several times how to make sure that Φ_e^G is recursive. To do this we want a be in a situation in which there are no extensions of the current approximation that e-split.

nosplits **Lemma 9.2.11** *If T is a partial tree such that there are no σ and τ such that $T(\sigma)|_e T(\tau)$, $G \in [T]$ and Φ_e^G is total then Φ_e^G is recursive.*

Proof. As usual, to compute $\Phi_e^G(x)$ we search for any σ such that $\Phi_e^{T(\sigma)}(x)\downarrow$. Since $\Phi_e^G(x)\downarrow$ there is an initial segment γ of G such that $\Phi_e^\gamma(x)\downarrow = \Phi_e^G(x)$. As $G \in [T]$ there is a τ such that $\gamma \subseteq T(\tau) \subset G$ and so τ is a string as desired. We then note that, for any such σ, $\Phi_e^{T(\sigma)}(x) = \Phi_e^{T(\tau)}(x) = \Phi_e^G(x)$ as otherwise $T(\sigma)|_e T(\tau)$. ■

We must now argue that if we cannot extend a given T to one with no e-splits on it as above, then we can guarantee that, if total, $\Phi_e^G \geq_T G$. To this end, we define another operation on trees that proceeds by searching for e-splits.

esplittree **Definition 9.2.12** *The* e-splitting subtree*,* $Sp(T,e)=S$*, of a partial recursive tree* T *is defined by recursion.* $S(\emptyset)=T(\emptyset)$*. If* $S(\sigma)=T(\tau)$ *then we search for* $\tau_0,\tau_1\supseteq\tau$ *such that the* $T(\tau_i)$ *e-split. We let* τ_0 *and* τ_1 *be the first such pair found in a standard search and set* $S(\sigma\hat{\ }i)=T(\tau_i)$*. A partial recursive tree* S *is an* e-splitting tree *if, for every* σ*, if one of* $S(\sigma\hat{\ }0)$*,* $S(\sigma\hat{\ }1)$ *is convergent then both are and they form an e-split.*

esplits **Proposition 9.2.13** *$Sp(T,e)$ is a partial recursive subtree of T with an index given uniformly recursively in one for T. If $Sp(T,e)$ is not total then there is a τ such that there are no e-splits on T above τ for some τ. Indeed, if $Sp(T,e)(\hat{\tau})\downarrow$ but $Sp(T,e)(\hat{\tau}\hat{\ }0)\uparrow$ and $T(\tau)=Sp(T,e)(\hat{\tau})$, then there are no e-splits on T above τ. Moreover, for any σ, $Sp(T,e)(\sigma\hat{\ }0)\downarrow\Leftrightarrow Sp(T,e)(\sigma\hat{\ }1)\downarrow$ and so $Sp(T,e)$ is an e-splitting tree.*

Proof. The assertions about the uniformity of the procedure of forming the e-splitting subtree and the equiconvergence of $Sp(T,E)(\sigma\hat{\ }i)$ for $i\in\{0,1\}$ are immediate from the definition. As for the rest, if $S(\emptyset)\uparrow$ then $T(\emptyset)\uparrow$ and we are done trivially. Otherwise, let τ be such that $Sp(T,e)(\tau)\downarrow=T(\rho)$ for some ρ but $Sp(T,e)(\tau\hat{\ }i)\uparrow$ for some (equivalently both) $i\in\{0,1\}$. If there were an e-splititng on T above ρ then we would have $S(\tau\hat{\ }i)\downarrow$ for both $i\in\{0,1\}$ by definition. ■

Thus to satisfy the minimality requirement M_e, it suffices to prove that if T_σ has e-splits for every σ (and so we cannot use Lemma nosplits 9.2.11 to force Φ_e^G to be recursive if total) then $Sp(T,e)$ forces $G\leq_T\Phi_e^G$ if the latter is total.

complemma **Lemma 9.2.14 (Computation Lemma)** *If S is a partial recursive e-splitting tree, $G\in[S]$ and Φ_e^G is total then $G\leq_T\Phi_e^G$.*

Proof. We compute an ascending sequence γ_n of initial segments of G (and so G itself) from Φ_e^G by recursion. We begin with $\gamma_0=S(\emptyset)$ which is an initial segment of G since $G\in[S]$. Suppose we have $\gamma_n=S(\sigma_n)\subset G$. As $G\in[S]$, one of $S(\sigma_n\hat{\ }0)$ and $S(\sigma_n\hat{\ }1)$ is also an initial segment of G. Thus $S(\sigma_n\hat{\ }0)$ and $S(\sigma_n\hat{\ }1)$ are both convergent and e-split. We may then recursively find an x on which $\Phi_e^{S(\sigma_n\hat{\ }0)}(x)\downarrow\neq\Phi_e^{S(\sigma_n\hat{\ }1)}(x)\downarrow$. Exactly one of these two agrees with $\Phi_e^G(x)$. We choose that $i\in\{0,1\}$ and set $\sigma_{n+1}=S(\sigma_n\hat{\ }i)$. ■

We have thus proven the density of conditions needed to satisfy the minimality requirements.

itsoresplit **Lemma 9.2.15** *The sets $C_e=\{T|$ either there are no e-splits on T or T is an e-splititng tree$\}$ are dense in $\mathcal{S}$. Moreover, there are density functions for these sets which are uniformly recursive in $0''$ on the set of (recursive binary function) trees.*

Proof. By the above Lemmas, either there is a σ such that $Fu(T,\sigma)$ has no e-splits or $SP(T,e)$ is a total tree and so the C_e are dense. As the two options are Σ_2^0 and Π_2^0 properties, respectively, $0''$ can decide which option to take and, if the first is chosen then even $0'$ can find a suitable σ as there being no e-splits on T_σ is a Π_1^0 property. If the second is chosen then the index is given recursively. ■

spmindeg **Theorem 9.2.16** *There is a minimal degree* $\mathbf{g} \leq \mathbf{0}''$.

Proof. Take any generic sequence $\langle T_n \rangle$ meeting all the D_e and C_e. The associated generic set $G = \cup T_n(\emptyset)$ is of minimal degree. By the above results on the complexity of the density functions we may take the sequence and so G to be recursive in $0''$. ■

One naturally asks at this point if we can do better in terms of the complexity of the minimal degree we construct. The most obvious question is whether we can produce one below $\mathbf{0}'$. It seems clear that we cannot use Spector forcing for this as the notion of forcing (indeed even the set of conditions) is of degree $\mathbf{0}''$. Given the work that we have already done, however, one would try to use partial recursive trees instead. The basic lemmas that we have already proven (nosplits 9.2.11 and complemma 9.2.14) still work. The problem is that once we hit a partial tree, there may be no further extensions. We construct a sequence of trees that satisfy all the requirements and construct a minimal degree below $\mathbf{0}'$ in the next section. The crucial new facet of the construction is that we use partial trees but when we discover we have reached a terminal point we backtrack and revise the previous trees in our sequence. A priority argument is then needed to show that the sequence stabilizes and so we satisfy each requirement.

Another improvement that we can deal with in the setting of Spector forcing is saying something about the double jump of G. In particular, we can show that $G'' \equiv_T 0''$. As we have often seen, we can either introduce new dense sets (requirements) that directly control the double jump or cleverly argue that we have already done so. We present a direct proof an leave the indirect one as an exercise. The idea here is that $G'' \equiv_T Tot^G = \{e | \Phi_e^G \text{ is total}\}$ and so we want conditions that decide if $e \in Tot^G$. The route is similar to that taken to splitting trees. The first alternative is that we have a tree T and an x such that $\Phi_e^{T(\sigma)}(x) \uparrow$ for every σ. Obviously in this situation we have forced that $\Phi_e^G(x) \uparrow$ and so it is not total. The second alternative is to produce a tree T such that $\Phi_e^G(x) \downarrow$ for every x and every $G \in [T]$. The analog of the $Sp(T, e)$ is $Tot(T, e)$:

totdef **Definition 9.2.17** *If T is a (partial) tree then $S = Tot(T, e)$ is defined by recursion beginning with $S(\emptyset) = T(\emptyset)$. If we have $S(\sigma) = T(\tau)$ then search for a $\rho \supseteq \tau$ such that $\Phi_e^{T(\rho)}(|\sigma|) \downarrow$. If there is one we let ρ be the first found in a standard search and set $S(\sigma\hat{\ }i) = T(\rho\hat{\ }i)$ for $i \in \{0, 1\}$.*

Proposition 9.2.18 *An index for $Tot(T, e)$ can be found uniformly recursively in one for T. If $Tot(T, e)$ is not total then there is a σ and an x such that $\Phi_e^{T(\rho)}(x) \uparrow$ for every $\rho \supseteq \sigma$.*

Proof. This is immediate from the definition of $Tot(T, e)$. ■

totdense **Proposition 9.2.19** *The sets $B_e = \{T | \exists x \forall \sigma (\Phi_e^{T(\sigma)}(x) \uparrow)$ or $(\forall \sigma)(\forall x < |\sigma|)(\Phi_e^{T(\sigma)}(x) \downarrow$ $)\}$ are dense in $\mathcal{S}$ and uniformly recursive in $0''$ and so have density functions uniformly recursive in $0''$.*

Proof. To see that the B_e are dense consider any T. If $Tot(T,e)$ is a total function we have the desired extension. If not, then there is a σ and an x such that $\Phi_e^{T(\rho)}(x)\uparrow$ for every $\rho \supseteq \sigma$. So T_σ is then the desired extension. That the B_e are uniformly recursive in $0''$ is immediate from their definition. ∎

Proposition 9.2.20 *If G is a generic defined from a sequence $\langle T_n\rangle \leq_T 0''$ meeting all the B_e then $G'' \equiv_T 0''$.*

Proof. To decide if $e \in Tot^G \equiv_T G''$, find an s such that $T_s \in B_e$ and see which clause of the definition of B_e is satisfied by T. ∎

mintot **Theorem 9.2.21** *There is a minimal degree* $\mathbf{g}$ *with* $\mathbf{g}'' = \mathbf{0}''$.

Proof. Add the dense sets B_e to those C_e and D_e considered before. There is a generic sequence recursive in $0''$ meeting all these sets and the generic G associated with it has all the desried properties. ∎

Of course, as might be naively expected, functions of minimal degree cannot have any strong domination properties. For example, none can be $\overline{\mathbf{GL}}_2$ by Theorem ngl2genseq ??. Even more striking is the fact that there is a single function of degree $\mathbf{0}'$ that dominates every function of minimal degree. This follows from the proof of Theorem ngl2genseq ??. In particular, by Proposition anreq 8.4.3 and Theorem anrgenseq 8.4.5 m_K, the least modulus function for $0'$ is such a function. For the minimal degrees we have constructed so far, we can say even more.

Exercise 9.2.22 *Show that the minimal degree constructed in Theorem mintot 9.2.21 is* $\mathbf{0}$*-dominated, i.e. every function recursive in G is dominated by a recursive function.*

Exercise 9.2.23 *Show that the G constructed in Theorem mintot 9.2.21 has minimal tt and wtt degree. (Hint: Recall Exercise 0domtt 8.1.2.)*

Exercise 9.2.24 *Show that the minimal degree constructed in Theorem spmindeg 9.2.16 has double jump* $\mathbf{0}''$. *Hint: show that meeting the dense sets C_e guarantees that the sequence meets the B_e as well.*

posnerl **Exercise 9.2.25 (Posner's Lemma)** *Show that meeting the dense sets C_e also guarantees that a generic sequence meets the D_e as well. Hint: Consider an n such that, for every σ and z, $\Phi_n^\sigma(z) = 0$ if $\neg(\exists x < |\sigma|)(\sigma(x) \neq \Phi_{e,|\sigma|}(x)\downarrow)$ and $\Phi_n^\sigma(z) = \sigma(z)$ otherwise.*

Exercise 9.2.26 *Show that for every* $\mathbf{d} > \mathbf{0}$ *there is a minimal degree* $\mathbf{g} \leq_T \mathbf{d}' \vee \mathbf{0}''$ *such that* $\mathbf{g} \not\leq_{\mathbf{T}} \mathbf{d}$. *??Improvement in Exercise ??*

treeofmin **Exercise 9.2.27** *There are continuum many minimal degrees. Indeed, there is a binary function tree $T \leq_T 0''$ such that every $G \in [T]$ is of minimal degree. Hint: Use conditions (T,n) where T is a (recursive binary function) tree, $n \in \mathbb{N}$ and extension is defined by $(S,m) \leq (T,n)$ if $S \leq_S T$, $m \geq n$ and $S(\sigma) = T(\sigma)$ for every σ of length $\leq n$.*

Exercise 9.2.28 *Show that in the previous exercise we may also guarantee that $G'' \equiv_T G \vee 0''$ for every $G \in [T]$.*

Exercise 9.2.29 *Show that for every $\mathbf{c} \geq \mathbf{0}''$ there is a minimal degree $\mathbf{g}$ with $\mathbf{g}'' = \mathbf{c} = \mathbf{g} \vee \mathbf{0}''$.*

Definition 9.2.30 *A binary tree T is* pointed *if every $A \in [T]$ computes T. It is* uniformly pointed *if there is an e such that $\Phi_e^A = T$ for every $A \in [T]$.*

Exercise 9.2.31 *Relativize Theorem $\overset{\texttt{spmindeg}}{9.2.16}$ to an arbitrary degree $\mathbf{c}$ to prove that every degree $\mathbf{c}$ has a* minimal cover*, i.e. a $\mathbf{g} > \mathbf{c}$ such that the open interval $(\mathbf{c}, \mathbf{g})$ is empty. Hint: One can proceed as usual by adding a $C \in \mathbf{c}$ into all oracle computations or one can use uniformly pointed trees recursive in C. In this case, just use binary function trees recursive in C which are subtrees of the tree T defined by $T(\sigma)(2n) = C(n)$ and $T(\sigma)(2n+1) = \sigma(n)$.*

Exercise 9.2.32 *All of the other results of this section now relativize.*

Exercise 9.2.33 *Prove that every strictly ascending sequence of degrees has a minimal upper bound $\mathbf{g}$. Hint: If the given sequence is $\mathbf{c}_n$, use uniformly pointed trees of degree $\mathbf{c}_n$ for some n.*

Exercise 9.2.34 *Show that the $\mathbf{g}$ of the previous exercise can be constructed so that $\mathbf{g}'' \leq \oplus \mathbf{c}_n''$.*

Exercise 9.2.35 *Show that one can also get two least upper bounds $\mathbf{g}_0$ and $\mathbf{g}_1$ for the $\mathbf{c}_n$ of the previous exercise with $(\mathbf{g}_0 \vee \mathbf{g}_1)'' \leq \oplus \mathbf{c}_n''$. Note that these $\mathbf{g}_i$ form an exact pair for the ideal generated by the $\mathbf{c}_n$.*

Exercise 9.2.36 *Thus if in the previous two exercises $\mathbf{c}_n = \mathbf{0}^{(n)}$ then one gets a minimal upper bound $\mathbf{g}$ for the $\mathbf{0}^{(n)}$ such that $\mathbf{g}'' = \mathbf{0}^{(\omega)}$ and indeed two such (which then form an exact pair for the arithmetic degrees) with $(\mathbf{g}_0 \oplus \mathbf{g}_1)'' = \mathbf{0}^{(\omega)}$.*

Exercise 9.2.37 *Prove that there is a tree T such that each path on T is a minimal upper bound for the ascending sequence $\mathbf{c}_n$.*

Definition 9.2.38 *A tree T is a* delayed e-splitting tree *if for every n there is an $m > n$ such that the strings $T(\sigma)$ for $|\sigma| = m$ are pairwise e-splititng.*

Exercise 9.2.39 *Prove the computation lemma for delayed e-splitting trees.*

Exercise 9.2.40 *Uniform trees; strongly uniform = 1-trees. one every path of minimal degree, $F : \mathbb{N} \rightarrow \{0, 1, 2\}$. minimal degrees generate $\mathcal{D}$ minimal m-degree Perhaps write out??*

Exercise 9.2.41 *other applications??*

sacksmin
9.3 Partial trees and Sacks minimal degrees

Sacksmin **Theorem 9.3.1 (Sacks)** *There is a minimal degree below* $0'$.

Our plan is to use partial recursive binary trees in a construction recursive in $0'$. We have already seen (Lemmas nosplits 9.2.11, complemma 9.2.14, nosplitsoresplit 9.2.15 and Proposition esplits 9.2.13 that we can handle both the diagonalization and minimality requirements by using subtrees of the form $Fu(T,\sigma)$ and $Sp(T,e)$ even if they are partial as long as we do not run into a node with no convergent extensions on the trees we are using. Now $0'$ can recognize this situation when it occurs. Thus the problem is what to do when we arrive at a node with no extensions on a tree. Of course, we must change the tree we intend our set to be on but we must do so in a way that eventually stabilizes so that, for each requirement, we remain, from some point onward, on some partial tree that satisfies the requirement.
Proof. At stage s, we will have already specified an initial segment α_s of the set A of minimal degree that we are building and a sequence (of indices for) nested partial recursive trees $T_{0,s} \geq_S T_{1,s} \geq_S \cdots \geq_S T_{k_s,s}$ with α_s on each of them (i.e. there are $\sigma_{i,s}$ such that $T_{i,s}(\sigma_{i,s}) = \alpha_s$). In fact, we will have $\alpha_s = T_{k,s}(\emptyset)$. $T_{0,0}$ is the identity function on binary strings. (Indeed, as will become clear, $T_{0,s}$ is the identity function for every s.) Each $T_{i+1,s}$ will be either $Sp(Fu(T_{i,s},j),i)$ for some $j \in \{0,1\}$or $Fu(T_{i,s},\sigma)$ for some σ and will be devoted to satisfying the minimality requirement for Φ_i with the choice of j devoted to satisfying the diagonalization requirements.

We now find the least $i \leq k_s$ such that $T_{i,s}(\sigma_{i,s}\hat{\ }0)\uparrow$. Let $k_{s+1} = i$ if one such exists, and let $k_{s+1} = k_s + 1$ otherwise. Note that this can be done recursively in $0'$ as we have indices for each $T_{i,s}$ as a partial recursive function.

- In the first case, we know that $T_{i,s}(\sigma_{i,s}\hat{\ }0)\uparrow$ while $T_{i-1,s}(\sigma_{i-1,s}\hat{\ }0)\downarrow$. Note that in this case T_i obviously cannot be of the form $Fu(T_{i-1},\sigma)$ and so (by the rules of the construction which we are maintaining by induction) must be of the form $Sp(Fu(T_{i,s},j),i)$. Thus by Proposition esplits 9.2.13 there are no extensions of α_s on T_{i-1} which i-split. We now let $T_{k_s+1} = Fu(T_{i-1,s},\sigma_{i-1.s}\hat{\ }0)$ (with the intention that we will satisfy the minimality requirement for Φ_i by being on a tree with no i-splits).

- In the second case, we let $T_{k_s+1,s+1} = Sp(Fu(T_{k_s,s},j),k_s)$ where we choose j so that $\Phi_{k_s} \neq T_{k_s+1,s+1}(\emptyset)$ (to be specific, say we choose $j = 1$ if $\exists x(T_{k_s,s}(1) \neq \Phi_{k_s}(x)\downarrow)$ and $j = 0$ otherwise) and with the hope that we will remain on this tree and so satisfy the minimality requirement for Φ_{k_s} by being on a k_s-splitting tree.

- In either case, we let $T_{i,s+1} = T_{i,s}$ for $i < k_{s+1}$ and $\alpha_{s+1} = T_{k_{s+1},s+1}(\emptyset)$. The trees T_i are, of course, not defined at $s+1$ for $i > k_{s+1}$.

We now claim that the $T_{i,s}$ stabilize, i.e. there is a tree $T_i = \lim_{s\to\infty} T_{i,s}$ and all the requirements are satisfied. Note that if $T_{i,s}$ reaches its limit by stage t then $k_s > i$ for $s > t$. Suppose, by induction, that $T_{i,s}$ first reaches its limit T_i at stage s. At $s+1$ we set $T_{i+1,s+1} = Sp(Fu(T_{i,s},j),i)$ (for some j) and we satisfy the diagonalization requirement for Φ_i. If we never change $T_{i+1,t}$ at a $t > s$ then $T_{i+1} = Sp(Fu(T_{i,s}, j), i))$ and we satisfy the minimality requirement for Φ_i by Lemma complemma 9.2.14. If there is a stage after s at which we first change T_{i+1}, i.e. $T_{i+1,t} \neq T_{i+1,t+1}$ it must be because we are in the first case at stage t and we set $k_{t+1} = i+1$ and $T_{i+1,t+1} = Fu(T_i, \sigma_{i,t}\hat{\ }0)$ because $Sp(Fu(T_{i,t}, j), i))(\sigma_{i+1,t}\hat{\ }0)$ is divergent. In this case, we can never change T_{i+1} again. (No smaller one ever changes by our choice of s and it can never be chosen as the least point of divergence as long as it is a full subtree of the previous tree.) Moreover, α_v remains on T_i on which there are no i-splits above α_t (Proposition esplits 9.2.13). Thus we satisfy the minimality requirement for Φ_i by Lemma nosplits 9.2.11. ■

Note that, in contrast to the Spector minimal degrees, no set recursive in $0'$ (and so even those of minimal degree) is $\mathbf{0}$-dominated by Theorem delta2dom 8.2.3. In ?? we will actually need to know a bit more about the set A of minimal degree that we have just constructed.

Corollary 9.3.2 *The set A of minimal degree constructed above is actually $\leq_{wtt} 0'$.*

Proof. To see that $A \leq_{wtt} 0'$ we need a recursive function f such that $f(n)$ bounds use from $0'$ needed to compute $A(n)$. An abstract view of the above construction is that at each stage s we have a number $k_s \leq s+1$ and a sequence of indices for partial trees $T_{i,s}$ for $i \leq k_s$. (Note that $\alpha_s = T_{k_s,s}(\emptyset)$.) We then ask for each $i \leq k_s$ if $T_{i,s}(\sigma_{i,s}\hat{\ }0)\downarrow$ where this question is equivalent to the one that asks if $\exists\tau(T_{i,s}(\tau) = T_{k_s,s}(\emptyset)\ \&\ T_{i,s}(\tau\hat{\ }0)\downarrow)$. Each possible set of answers to these questions determines $0 < k_{s+1} \leq k_s + 1 \leq s+1$ and the indices for the $T_{i,s+1}$ for $i \leq k_{s+1}$ except when they say that $k_{s+1} = k_s + 1$. In this case, we need to ask one more question of $0'$: is there an x such that $T_{k_s,s}(1) \neq \Phi_{k_s}(x)\downarrow$? Thus we can recursively lay out all possible routes of the construction as a tree which at level s is (at most) $s+1$ branching along with the (at most $s+1$ many) questions of $0'$ needed to determine at each node of the tree at level s what stage $s+1$ of the construction would be if the given node corresponds to the actual stage s of the construction. Now to compute $A(n)$ note that we extend α_s at every stage of the construction so we only need a recursive bound on the questions asked in any possible run of the construction for n many stages. As the indices for all the possible $T_{i,s}$ are uniformly computable from the various assumed answers at the previous stages, it is clear that there is a recursive bound on the questions that are needed in all possible runs of the construction for n many steps. ■

Exercise 9.3.3 *Theorem Sacksmin 9.3.1 and the Corollary above relativize to arbitrary degrees $\mathbf{c}$ to give a minimal cover $\mathbf{g}$ of $\mathbf{c}$ with $\mathbf{g} \leq_{wtt} \mathbf{c}'$.*

Exercise 9.3.4 *??Show that for every $\mathbf{d} > \mathbf{0}$ there is a minimal degree $\mathbf{g} \leq_T \mathbf{d} \vee \mathbf{0}'$ such that $\mathbf{g} \not\leq_{\mathbf{T}} \mathbf{d}$. Hint my construction in L p. 192?? only for $\mathbf{d} < \mathbf{0}'$??otherwise below $\mathbf{d}'$??*

0'treeofmin **Exercise 9.3.5** *Construct a tree $T \leq_T 0'$ such that every path on T is of minimal degree.*

Cone avoiding?? join ?? Complementation??

9.4 Minimal degrees below degrees in $\mathbf{H}_1$ and $\mathbf{GH}_1$

We want to prove that if $\mathbf{h} \in \mathbf{GH}_1$ then there is a minimal degree $\mathbf{a} < \mathbf{h}$. The proof builds on the construction of a Sacks minimal degree with highness giving us an approximation to but is unusual in that it relies on the recursion theorem to make the approximations work.

Remark: Not below every $\mathbf{H}_2$ Lerman [??]).

Question 9.4.1 *If $A >_T 0$ is r.e. then there is a minimal degree below A [??]. Can one construct such a degree with the techniques presented in this chapter and the previous one or some variation of them?*

cone avoiding, join, complementation results?

9.5 Jumps of minimal degrees

At the end of §spectormin 9.2 we analyzed the possible double jumps of Spector minimal degrees. In this section we want to investigate the possible single jumps of arbitrary minimal degrees. Note first that every minimal degree is $\mathbf{GL}_2$ because every $\overline{\mathbf{GL}_2}$ degree has a 1-generic degree below it by Theorem gl2lgen 8.3.3. We will see that there are minimal degrees in both $\mathbf{GL}_1$ and $\mathbf{GL}_2 - \mathbf{GL}_1$. Finally, we will completely characterize the jumps of minimal degrees by giving a new proof due to Lempp, J. Miller S. Ng and L. Yu of Cooper's jump inversion theorem that every $\mathbf{c} \geq \mathbf{0}'$ is the jump of a minimal degree. The situation below $\mathbf{0}'$ is more complicated. While there are both $\mathbf{L}_1$ and $\mathbf{L}_2 - \mathbf{L}_1$ minimal degrees, not every degree $\mathbf{c}$ which is r.e. in and low over $0'$ is the jump of a minimal degree below $\mathbf{0}'$ (refs?? Shore noninversion theorem, Cooper).

9.5.1 Narrow trees and $\overline{\mathbf{GL}_1}$ minimal degrees

To produce a minimal degree not in $\mathbf{GL}_1$ we must combine a diagonalization of A' against $\Phi_e(A \oplus 0')$. The key idea here are the narrow subtrees $N(T)$.

Definition 9.5.1 *The* narrow subtree *$N(T)$ of a total tree T is defined by recursion. $N(T)(\emptyset) = T(\emptyset)$. If $N(T)(\sigma) = T(\tau)$ then $N(T)(\sigma\hat{\ }i) = T(\tau\hat{\ }0\hat{\ }i)$.*

Proposition 9.5.2 *If T is recursive so is $N(T)$ and an index for it can be found uniformly recursively in one for T. Of course, as with any recursive tree the question of whether $A \in [N(T)]$ is Π^0_1 in A and the index for $N(T)$ and so uniformly recursive in A', i.e. there is a recursive f such that $(\forall A)(A \in [N(T)] \Leftrightarrow f(n) \in A')$ where n is any index for $N(T)$.*

Our plan is to use narrow subtrees to diagonalize. Intuitively we stay on some $N(T)$ with index i until we see that $\Phi_e^{\alpha_s \oplus 0'}(f(i)) \downarrow = 1$. At that point we will make A go off $N(T)$ and so guarantee that $A' \neq \Phi_e^{A\oplus 0'}$. Formally we prove that diagonalization is dense.

nardense **Lemma 9.5.3** *The sets $F_e = \{T | (\forall G \in [T]) \neg(\Phi_e^{A\oplus 0'} = A')\}$ are dense in the Spector notion of forcing and there is a density function which is uniformly recursive in $0'$ on (the indices for) recursive trees..*

Proof. Let n be an index for T and consider $N(T) = S$. If there is a σ such that $\Phi_e^{S(\sigma)\oplus 0'}(f(n)) \downarrow = 1$ then the desired extension $\hat{T}$ of T is $Fu(T, \tau\hat{\ }1)$ where $T(\tau) = S(\sigma)$. The point here is that no $A \in [\hat{T}]$ is on $S = N(T)$ while $\Phi_e^{A\oplus 0'}(f(n)) \downarrow = 1$ for every $A \in \hat{T}$ and so $\Phi_e^{A\oplus 0'} \neq A'$. On the other hand, if there is no such σ then $N(T)$ is the desired extension of T as $f(n) \in A'$ for every $A \in [N(T)]$ while $\neg(\Phi_e^{A\oplus 0'}(f(n)) = 1)$ for every $A \in [N(T)]$ by our case assumption. It is clear that finding the desired extension of T is recursive in $0''$. ■

Theorem 9.5.4 *There is a minimal degree $\mathbf{g} \leq \mathbf{0}''$ with $\mathbf{g} \notin \mathbf{GL}_1$. We may also guarantee that $\mathbf{g}'' = \mathbf{0}''$.*

Proof. Simply add the dense sets F_e to the ones D_e and C_e in the proof of Theorem spmindeg 9.2.16 to be met in the construction of G. To guarantee that $\mathbf{g}'' = \mathbf{0}''$ add in the dense B_e of Proposition totdense 9.2.19. ■

Exercise 9.5.5 *Modify the proof of Theorem sacksmin 9.3 to construct an $A \leq_T 0'$ of minimal degree with degree not in $\mathbf{L}_1$. Hint: intersperse stages at which one puts $T_{i+1,s+1} = N(T_{i,s})$ and then stays in this tree until $\Phi_e^{\alpha_s}(f(n) \downarrow = 1$ where e and n are as in Lemma nardense 9.5.3 for $T_{i,s}$.*

9.5.2 Cooper's jump inversion theorem

We want to prove that every degree $\mathbf{c} \geq \mathbf{0}'$ is the jump of a minimal degree. To do this we modify the definition of the e-splitting subtree in an attempt to force the jump when we can.

ejumpspl **Definition 9.5.6** *The e-**jump splitting subtree of** T, $JSp(T, e) = S$ is defined by recursion. $S(\emptyset) = T(\emptyset)$ which is labeled ω. Suppose $S(\sigma) = T(\tau)$ is defined and is labeled some $m \leq \omega$. We search simultaneously for $\tau_0, \tau_1 \supseteq \tau$ such that $T(\tau_0)|_e T(\tau_1)$ and for a $\rho \supseteq \tau$ and an $n < m$ such that $\Phi_n^{T(\rho)}(n) \downarrow$ but $\Phi_n^{T(\tau)}(n) \uparrow$. If we first (in some canonical search order) find an e-split then we let $S(\sigma\hat{\ }i) = T(\tau_i)$ and label them both ω. If we first find a ρ and n as described we let $S(\sigma\hat{\ }0) = T(\rho)$ and label it n. $S(\sigma\hat{\ }1)$ is undefined in this case. (Of course, if neither search terminates, $S(\sigma\hat{\ }i) \uparrow$ for both $i = 0, 1$.)*

Proposition 9.5.7 *If T is (partial) recursive then so is $JSp(T, e)$ and an index for it can be found uniformly recursively in one for T.*

noisol **Lemma 9.5.8** *If $JSp(T,e) = S$, then there are no isolated paths on S, i.e. if $A \in [S]$ then there are infinitely many σ such that $S(\sigma) \subset A$ and $S(\sigma\hat{\ }i)\downarrow$ for $i = 0, 1$.*

Proof. This is immediate from the fact that whenever $S(\sigma)\downarrow$ but not both of $S(\sigma\hat{\ }i)$ are defined then only $S(\sigma\hat{\ }0)$ is defined and its label is in $\mathbb{N}$ and remains strictly decreasing until we reach a $\sigma\hat{\ }0^t$ such that both $S(\sigma\hat{\ }0^t\hat{\ }0)$ and $S(\sigma\hat{\ }0^t\hat{\ }1)$ are defined and their labels are ω. Thus we can continue to extend only one side (necessarily the 0 one) as we follow A on S only finitely often. ■

jcomplemma **Lemma 9.5.9** *If $S = JSp(T,e)$, $G \in [S]$ and Φ_e^G is total then $G \leq_T \Phi_e^G$.*

Proof. As for the basic Computation Lemma complemma 9.2.14, we compute an ascending sequence γ_n of initial segments of G (and so G itself) from Φ_e^G by recursion. We also compute σ_n and τ_n such that $T(\tau_n) = S(\sigma_n) = \gamma_n$ and its label m_n on S. We begin with $\gamma_0 = T(\emptyset) = S(\emptyset)$ which is an initial segment of G since $G \in [S]$. Suppose we have $\gamma_n = S(\sigma_n) = T(\tau_n) \subset G$ and m_n. As $G \in [S]$, one of $S(\sigma_n\hat{\ }0)$ and $S(\sigma_n\hat{\ }1)$ is also an initial segment of G. We follow the procedure given in the definition of $JSp(T,e)(\sigma_n\hat{\ }i)$. If we first find an e-split then both $S(\sigma_n\hat{\ }i)$ are convergent. As they e-split we can decide which one is an initial segment of G using Φ_e^G as in the basic Computation Lemma and continue our recursion. If instead, we first find a new convergence for $\Phi_{\hat{n}}^{T(\rho)}(\hat{n})$ for $\hat{n} < n$, only $S(\sigma_n\hat{\ }0)$ is defined and it is then the next initial segment γ_{n+1} of G as required. Of course, $\sigma_{n+1} = \sigma_n\hat{\ }0$. This also supplies us with the next τ_{n+1} and $m_{n+1} = \hat{n}$. ■

lowmin **Theorem 9.5.10** *There is an A of minimal degree with $A' \equiv_T 0'$.*

Proof. The construction is similar to that for Theorem Sacksmin 9.3.1 except that we use e-jump splitting subtrees instead of e-splitting subtrees and we have to be a bit more careful about how we go off the partial trees.

At stage s, we will have an already specified initial segment α_s of A and a sequence (of indices for) nested partial recursive trees $T_{0,s} \geq_S T_{1,s} \geq_S \cdots \geq_S T_{k_s,s}$ with α_s on each of them, indeed with $\alpha_s = T_{k,s}(\emptyset)$. $T_{0,s}$ is the identity function for every s. Each $T_{i+1,s}$ will be either $JSp(Fu(T_{i,s},\sigma), i)$ for some σ or $Fu(T_{i,s},\sigma)$ for some σ.

We begin our search for k_{s+1} with $T_{k_s,s}$. We ask if $T_{k_s,s}(1)\downarrow$. If it is, so is $T_{k_s,s}(0)$. We then set $T_{k_{s+1}} = JSP(Fu(T_{k_s}, j), k_s)$ where we choose j so that $\Phi_{k_s}(x) \neq T_{k_s,s}(j)(x)$ for some x and set $k_{s+1} = k_s + 1$. If $T_{k_s,s}(1)\uparrow$ we ask if $T_{k_s,s}(0)\downarrow$. If so we repeat our procedure with T_{k_s} replaced by $Fu(T_{k_s}, 0)$. By Lemma noisol 9.5.8 this process eventually terminates either with an m such that $T_{k_s}(0^m\hat{\ }1)\downarrow = Fu(T_{k_s}, 0^m)(1)\downarrow$ and so a definition of $k_{s+1} = k_s + 1$ and $T_{k_{s+1}} = JSP(Fu(T_{k_s}, 0^m\hat{\ }j), k_s)$ or an m such that $T_{k_s}(0^m)\downarrow$ but $T_{k_s}(0^{m+1})\uparrow$ (m could be 0 and we take $0^0 = \emptyset$). In the later case, we move to T_{k_s-1} beginning with the σ_1 such that $T_{k_s-1}(\sigma_1) = T_{k_s}(0^m)$ and asking if $T_{k_s-1}(\sigma_1\hat{\ }1)\downarrow$. Continuing in this way we eventually reach l and m such that $T_{l,s}(\sigma\hat{\ }0^m\hat{\ }j)\downarrow$ for some σ and each $j \in \{0,1\}$ as $T_{0,s}$ is always the identity function and so defined at $\sigma\hat{\ }1$ for every σ. We now let $k_{s+1} = l + 1$ and $T_{k_{s+1}} = Fu(T_{l,s}, \sigma\hat{\ }0^{m+1})$. We conclude the stage by setting $\alpha_{s+1} = T_{k_{s+1}}(\emptyset)$. Note that we extend α_s at every stage and $A = \cup\alpha_s \leq_T 0'$.

It is clear that the construction and so A is recursive in $0'$. We must now verify that the $T_{i,s}$ stabilize to trees T_i, all the requirements to make A of minimal degree and that $A' \leq_T 0'$. We argue much as in the proof of Theorem Sacksmin 9.3.1 for the first two claims:

Note again that if $T_{i,s}$ reaches its limit by stage t then $k_s > i$ for $s > t$. Suppose, by induction, that $T_{i,s}$ first reaches its limit T_i at stage s. At $s+1$ we set $T_{i+1,s+1} = JSP(Fu(T_{i_s}, 0^m\hat{\ }j), i)$ for some m and j as the only other possibilities change T_i. This action satisfies the diagonalization requirement for Φ_i. If we never change $T_{i+1,t}$ at a $t > s$ then $T_{i+1} = JSP(Fu(T_i, 0^m\hat{\ }j), i))$ and we satisfy the minimality requirement for Φ_i by Lemma jcomplemma 9.5.9. If there is a first stage after s at which we change T_{i+1}, i.e. $T_{i+1,t} \neq T_{i+1,t+1}$, then it must be that we reached a situation with $T_{l,t}(\sigma\hat{\ }0^{\hat{m}}\hat{\ }\hat{j}) \downarrow$ for some σ and both $\hat{j} \in \{0,1\}$ with l the first such we find in our search starting with k_t and moving downward and $\hat{m}$ the least such for l. As we now redefine T_{l+1} it must be that $l = i$ by our induction hypothesis. As t is the first stage after s at which we change T_{i+1}, $T_{i+1,t} = JSP(Fu(T_{i,s}, 0^m\hat{\ }j), i)$. As we did not end our search for this l with $l+1 = i+1$, if $T_{l,t}(\sigma) = T_{i+1,t}(\tau)$ then $T_{i+1,t}(\tau\hat{\ }0) \uparrow$. By the definition of $T_{i+1,t} = JSP(Fu(T_{i,s}, 0^m\hat{\ }j), i)$ this means that there are no i-splits on $T_{i,s} = T_i$ above σ. As $A \in [Fu(T_i, \sigma)]$ we satisfy the minimality requirement for Φ_i by Lemma nosplits 9.2.11. Once T_{i+1} is a full subtree of T_i (as it is at $t+1$), it can never be changed again as that would change some T_k for $k \leq i$ contrary to our choice of $s < t$.

To compute A' from $0'$ find a stage of the construction s at which we end the construction with $l \leq k_s$ and $T_{l,s}(\sigma\hat{\ }0^m\hat{\ }j) \downarrow$ for $j \in \{0,1\}$ and we let $k_{s+1} = l+1$ and $T_{l+1,s+1} = Fu(T_{l,s}, \sigma\hat{\ }0^{m+1})$. In this case we have $T_{l+1,s}(\tau\hat{\ }0) \uparrow$ where $T_{l+1,s}(\tau) = T_{l,s}(\sigma)$. If n is the label of $T_{l+1,s}(\tau)$, this means that there is no extension ρ of $T_{l+1,s}(\tau)$ on $T_{l,s}$ such that $\Phi^{\rho}_{\hat{n}}(\hat{n}) \downarrow$ but $\Phi^{T_{l+1,s}(\tau)}_{\hat{n}}(\hat{n}) \uparrow$ for $\hat{n} < n$. We now claim that, for $\hat{n} < n$, $\hat{n} \in A \Leftrightarrow \Phi^{T_{l+1,s}(\tau)}_{\hat{n}}(\hat{n}) \downarrow$. As long as α_t stays on $T_{l,s}$ for $t > s$ (as it is now) the claim is obvious. The only way α_t can leave $T_{l,s}$ for the first time after s at t is for the same situation to occur with $l_1 < l$. In this case, the associated label must be $n_1 \geq n$ (as no new convergences below n can occur as long as we remain on $T_{l,s}$). In this case, no new convergences below n_1 can occur as long as we remain on $T_{l_1,t}$. This process must halt and so we eventually stay on some tree $T_{\hat{l},\hat{t}}$ on which there are no new convergences below some $\hat{n} \geq n$. To see that our original search in this procedure must find such stages s with arbitrarily large n, fix an r and start with a stage u by which $\forall e \leq r(\Phi^A_e(e) \downarrow \Leftrightarrow \Phi^{\alpha_u}_e(e) \downarrow)$. Now consider a $v > u$ for which Φ_v is the empty function. When we reach the first stage w at which $k_w = v+1$ for the first time after T_i has reached its limit for $i \leq v$ we set $T_{v+1,w+1} = JSP(Fu(T_{v,w}, 0^m\hat{\ }j), v))$ for some m and j. This tree has $T_{v+1,w+1}(\tau\hat{\ }1) \uparrow$ for every τ and so we would act as described above and for an $n \geq r$. ■

Theorem 9.5.11 *There is a binary function tree $T \leq_T 0'$ such that every $A \in [T]$ is of minimal degree and, moreover, $A' \equiv_T A \vee 0'$.*

Proof. We define T by recursion beginning with $T(\emptyset)=\emptyset$. Along each path in T we are using the construction of Theorem lowmin 9.5.10 with the change that when we would have chosen one $j\in\{0,1\}$ and set $T_{k_s+1}=Fu(S,j)$ for some S we follow both possibilities and define the next branching in T as the result of the two choices of j in the original construction. Thus at any node ρ when we have $T(\rho)$ defined we have an associated run of the above construction during which we have chosen $j=\rho(m)$ at the mth instance where we had to choose a j in the construction. To define $T(\rho\hat{\ }i)$ we now continue the construction as in the previous theorem until we reach the next stage s at which we must choose a j and set $T_{k_s+1}=JSp(Fu(T_{k_s,s},j),k_s)$. We now let $T(\rho\hat{\ }j)=JSp(Fu(T_{k_s,s},j),k_s)(\emptyset)$ for $j\in\{0,1\}$ and associate the version of the above construction in which we choose j with $T(\rho\hat{\ }j)$. ■

Corollary 9.5.12 (Cooper's Jump Inversion Theorem) *For every* $\mathbf{c}\geq\mathbf{0}'$ *there is a minimal degree* $\mathbf{a}$ *such that* $\mathbf{a}'=\mathbf{c}=\mathbf{a}\vee\mathbf{0}'$.

Proof. Take $C\in\mathbf{c}$ and let $A=\cup T(C\restriction n)$. ■

remark not all degrees REA in $\mathbf{0}'$ and low over it are jumps of minimal degrees below $\mathbf{0}'$ references.

Theorem lowmin 9.5.10 originally by Yates showed minimal below every nonrecursive r.e. degree and was already known (Theorem ??) that there are low nonrecursive r.e. degrees. Then ...

9.6 The minimal degrees generate $\mathcal{D}$

Our goal in this section is to prove that the minimal degrees generate $\mathcal{D}$ under join and meet. More specifically we will prove that for every $\mathbf{a}$ there are minimal degrees $\mathbf{m}_0$, $\mathbf{m}_1$, $\mathbf{m}_2$ and $\mathbf{m}_3$ such that $\mathbf{a}=(\mathbf{m}_0\vee\mathbf{m}_1)\wedge(\mathbf{m}_2\vee\mathbf{m}_3)$. Our forcing conditions in this section will all be recursive binary trees but we need a yet more restricted notion of tree. We begin with uniform trees (which will play a crucial role in the next chapter) and strongly uniform trees or 1-trees.

1treedef **Definition 9.6.1** *A binary tree T is* uniform *if for every n there are $\rho_{n.0},\rho_{n,1}\in 2^{<\omega}$ such that $T(\sigma\hat{\ }i)=T(\sigma)\hat{\ }\rho_{n,i}$ for every σ of length n. T is* strongly uniform *if, in addition, for every n, $\rho_{n.0}$ and $\rho_{n,1}$ are* adjacent*, i.e. there is exactly one j such that $\rho_{n.0}(j)\neq\rho_{n,1}(j)$. Strongly uniform trees are also called* 1-trees.

In this section all trees will be recursive 1-*trees* and they will be the conditions in our basic notion of forcing $\mathcal{P}$ with the usual notion of subtree as the extension relation. As $Fu(T,\sigma)$ is clearly a 1-tree for any 1-tree T, the diagonalization requirements D_e of Lemma spdiag 9.2.7 are still dense so we can meet those conditions as usual. Lemma nopslits ?? applies to any binary tree and so if our generic filter includes a tree with no e-splits then again, if Φ_e^G is total it is recursive. The computation lemma (complemma 9.2.14) also applies quite generally and so if the sets C_e of Lemma nosplitsoresplit 9.2.15 are dense then any generic for forcing with 1-trees will also be of minimal degree. Thus we must show that if the 1-tree T has no extensions without e-splits then it has an extension which is e-splitting. It is actually helpful in this setting to first provide the analog of $Tot(T, e)$ that forces totality and proves the density of the B_e of Lemma totdense 9.2.19.

1totdense **Lemma 9.6.2** *The sets $B_e = \{T | \exists x \forall \sigma (\Phi_e^{T(\sigma)}(x) \uparrow)$ or $(\forall \sigma)(\forall x < |\sigma|)(\Phi_e^{T(\sigma)}(x) \downarrow)\}$ are dense in $\mathcal{P}$.*

Proof. Given a 1-tree T we define a partial recursive function $S = Tot_1(T, e)$ by recursion beginning as usual with $S(\emptyset) = T(\emptyset)$. Let $\{\sigma_i | i < 2^n\}$ list all the strings of length n and assume that $S(\sigma_i) = T(\tau_i)$ has been defined for all $i < 2^n$. To define S for all ρ of length $n+1$, we search first for a μ_0 such that $\Phi_e^{T(\tau_0\hat{}\mu_0)}(n) \downarrow$. Then we recursively search for μ_i such that $\Phi_e^{T(\tau_0\hat{}\mu_0\hat{}\dots\hat{}\mu_i)}(n) \downarrow$. If we eventually find μ_i for all $i < 2^n$, then we let $\mu = \mu_0\hat{}\dots\hat{}\mu_{2^n-1}$ and set $S(\sigma_i\hat{}j) = T(\tau_i\hat{}\mu\hat{}j)$ for $j \in \{0, 1\}$. As T is a 1-tree it is easy to see that, if total, so is $Tot_1(T, e)$ and it satisfies the second clause of B_e. If it is not total then there is some n, τ_i and ν such that $T(\tau_i\hat{}\mu) \uparrow$ for every $\mu \supseteq \nu$. In this case, $Fu(T, \tau_i\hat{}\nu)$ satisfies the first clause of B_e with $x = n$. ■

We can now prove the remaining lemma that shows that all (sufficiently) generic G for $\mathcal{P}$ are of minimal degree.

itsoresplit **Lemma 9.6.3** *The sets $C_e = \{T | \ \exists x \forall \sigma (\Phi_e^{T(\sigma)}(x) \uparrow)$ or there are no e-splits on T or T is an e-splititng 1-tree$\}$ are dense in $\mathcal{P}$.*

Proof. By Lemma 1totdense 9.6.2, we may assume that the second clause of B_e is satisfied by T, i.e. $(\forall \sigma)(\forall x < |\sigma|)(\Phi_e^{T(\sigma)}(x) \downarrow)$. We may also assume that there is no extension of T that satisfies the second clause of C_e so we can find e-splits on any $R \subseteq T$. We now wish to define an e-splitting 1-tree $Sp_1(T, e) = S \subseteq T$. We begin with converting arbitrary e-splits into ones that are adjacent and then defining two new operations on 1-trees.

Claim 9.6.4 *For any $R \subseteq T$ there are adjacent σ and τ such that $R(\sigma)|_e R(\tau)$ and so, in particular, for any ρ there are $\sigma, \tau \supseteq \rho$ which are adjacent such that $R(\sigma)|_e R(\tau)$.*

Proof. By our second assumption on T there are μ and ν such that $R(\mu)|_e R(\nu)$. Without loss of generality we may take $|\mu| = |\nu| > n$ where $R(\mu)$ and $R(\nu)$ e-split at n. Consider then the sequence $\langle \sigma_i | i \leq k \rangle$ of adjacent binary strings of length n such that $\sigma_0 = \mu$ and $\sigma_k = \nu$. By our first assumption on $T \supseteq R$, $\Phi_e^{R(\sigma_i)}(n) \downarrow$ for every $i \leq k$. As the first and last of these have different values there must be an i such that $\Phi_e^{R(\sigma_i)}(n) \downarrow \neq \Phi_e^{R(\sigma_{i+1})}(n) \downarrow$. Our desired adjacent e-split is then given by $\sigma = \sigma_i$ and $\tau = \sigma_{i+1}$. ■

Definition 9.6.5 *For any tree R and $\mu \in 2^{<\omega}$ we define R^μ (the* transfer tree *of R over μ) for $|\nu| \leq |R(\emptyset)|$ as the tree such that, for every $\sigma \in 2$, $R^\mu(\sigma)$ is the string gotten from $R(\sigma)$ by replacing its initial segment of length $|\mu|$ by μ. For $R \subseteq T$ we define a new type of* *subtree $S = Sp_0(R, e)$. We begin by using the above Claim to construct sequences σ_i^0 and σ_i^1 for $i \in \mathbb{N}$ with σ_i^0 and σ_i^1 adjacent such that first, $R(\sigma_0^0)|_e R(\sigma_0^1)$ and then, in general, $R(\sigma_0^0\hat{}\cdots\hat{}\sigma_i^0\hat{}\sigma_{i+1}^0)|_e R(\sigma_0^0\hat{}\cdots\hat{}\sigma_i^0\hat{}\sigma_{i+1}^1)$. We now define S by recursion with $S(\emptyset) = R(\emptyset)$ and $S(\rho) = R(\Sigma\sigma_i^{\rho(i)})$ (where we use summation notation Σ for concatenation and the number of terms concatenated is $|\rho|$).*

Remark 9.6.6 *Note that as R is a 1-tree and the σ_i^0 and σ_i^1 are adjacent, S is also a 1-tree and, of course, $S \subseteq R$. Moreover, $S(\sigma)|_e S(\tau)$ for any $\sigma \neq \tau$ as the strings extend some e-split $R(\sigma_0^0)|_e R(\sigma_0^1)$ or $R(\sigma_0^0\hat{}\cdots\hat{}\sigma_i^0\hat{}\sigma_{i+1}^0)|_e R(\sigma_0^0\hat{}\cdots\hat{}\sigma_i^0\hat{}\sigma_{i+1}^1)$ for $i \geq 0$.*

■

Proof continued. We now define our e-splitting 1-tree $Sp_1(T, e) = S \subseteq T$ by recursion beginning with $S(\emptyset) = T(\emptyset)$. Let $\{\sigma_i | i < 2^n\}$ list all the strings of length n and assume that $S(\sigma_i) = T(\tau_i)$ has been defined for all $i < 2^n$. We let $R_0 = Sp_0(T_{\tau_0}, e)$ and for $0 < i < 2^n$ we let $R_i = Sp_0(R_{i-1}^{T(\tau_i)}, e)$ and $R = R_{2^n-1}$. We now let $S(\sigma_i\hat{}j) = R^{T(\tau_i)}(j)$. The verifications that this defines the next level of an e-splitting 1-tree contained in T are straightforward. By the definition of Sp_0, $|R(\emptyset)| = |R_i(\emptyset)| = |T(\tau_i)|$ for every $i < 2^n$ and $R(0)$ and $R(1)$ are adjacent. By the definition of the transfer trees, $R^{T(\tau_i)}(0)$ and $R^{T(\tau_i)}(1)$ are adjacent extensions of $T(\tau_i) = S(\sigma_i)$ and the extensions are given by the same pair of strings for each i as R is a 1-tree. Moreover, since $R \subseteq R_i$ for every $i < 2^n$, each $R^{T(\tau_i)}(j)$ is a node on $R_i = Sp_0(R_{i-1}^{\tau_i}, e)$ (where $R_{-1} = T$) and so by the Remark above, they form an e-splitting.

This completes the definition of level $n+1$ of S and so, by recursion of $S = Sp_1(T, e)$ which is an e-splitting 1-tree extending T as required to establish the density of the C_e.

■

We have now shown the forcing with 1-trees produces a minimal degree.

1treemin **Proposition 9.6.7** *Any generic meeting the dense sets B_e, C_e and D_e for forcing with 1-trees is of minimal degree.*

Exercise 9.6.8 *Show that there are generics G as in Proposition* 1treemin *9.6.7 with $G \leq_T 0''$ and indeed with $G'' \equiv_T 0''$.*

Exercise 9.6.9 *Show that the genric sets of Proposition* 1treemin *9.6.7 are of minimal m-degree.*

We next want a tree of such minimal degrees, i.e. a 1-tree T such that every path is of minimal degree. We move to a tree of trees as we did in Exercise treeofmin 9.2.27.

1treeofmin **Theorem 9.6.10** *If we force with the notion of forcing $\mathcal{P}_{1t}$ with conditions (T, n) where T is a 1-tree, $n \in \mathbb{N}$ and extension is defined by $(S, m) \leq_{\mathcal{P}_t} (T, n)$ if $S \leq_{\mathcal{S}} T$, $m \geq n$ and $S(\sigma) = T(\sigma)$ for every σ of length $\leq n$ and $V((T, n))$ is the finite binary 1-tree given by restricting T to strings of length n, then any sufficiently generic G is a 1-tree such that every path on G is of minimal degree.*

Proof. It is clear that G is a 1-tree from the fact that $V(P)$ is a 1-tree for every condition P and that if $Q \leq_{\mathcal{P}_1} P$ then $V(Q) \supseteq V(P)$ as 1-trees. To prove the theorem it suffices, by the last Proposition, to show that for each of the dense sets B_e, C_e and D_e any condition (T, n) there is a condition $(R, n) \leq_{\mathcal{P}_1} (T, n)$ such that for each σ of length n, $R_{R(\sigma)}$ is in the desired dense set. List the strings of length n as σ_i, $i < 2^n$. Begin with $S_0 = T_{T(\sigma_0)}$. By the relevant Lemma above (1totdensesplit 9.6.2, 1treesplit ?? and splitting 9.2.7) we can refine S_0 to a 1-tree S_1 which is in the dense set. We can then consider $S_1^{T(\sigma_1)}$ and refine it to S_2 which is also in the dense set. We continue in this way to define S_i for $i < 2^n$ by refining $S_i^{T(\sigma_i)}$ to get an S_{i+1} in the dense set. At the end we have $S = S_{2^n}$ such that $S^{T(\sigma_i)}$ is in the dense set for each $i < 2^n$. We now define R by $R(\rho) = T(\rho)$ for $|\rho| \leq n$ and for $\rho \supset \sigma_i$ $R(\rho) = S^{T(\sigma_i)}(\rho)$. It is clear that $(R, n) \leq_{\mathcal{P}_1} (T, n)$ and for each σ of length n, $R_{R(\sigma)}$ is in the desired dense set. Let G be a generic 1-tree meeting all these dense sets. Now any $M \in [G]$ is $\mathcal{P}$-generic for the previous notion of forcing with 1-trees to the extent required by Proposition 1treemin 9.6.7 and so is of minimal degree by that Proposition. ■

Exercise 9.6.11 *Show that there are generics G as in Theorem 1treeofmin 9.6.10 with $G \leq_T 0''$ and indeed with $G'' \equiv_T 0''$.*

Exercise 9.6.12 *For each $n \geq 3$, Show that there are sets A of minimal degrees which are Σ^0_n but not Δ^0_n. Hint: take a path in the G of Theorem 1treeofmin 9.6.10 which follows a path $C \in \Sigma^0_n - \Delta^0_n$, i.e. $A = \cup\{G(C \upharpoonright k) | k \in \mathbb{N}\}$. (For $n = 2$, the result can be proven using, among other things, Exercise 0'treeofmin 9.3.5.)??*

Finally, we use $\mathcal{P}_t$ to prove our main theorem for this section.

mingen **Theorem 9.6.13** *For every degree* $\mathbf{a}$ *there are minimal degrees* $\mathbf{m}_0$*,* $\mathbf{m}_1$*,* $\mathbf{m}_2$ *and* $\mathbf{m}_3$ *such that* $\mathbf{a} = (\mathbf{m}_0 \vee \mathbf{m}_1) \wedge (\mathbf{m}_2 \vee \mathbf{m}_3)$.

Proof. For any 1-tree G and set C, we let d_n be the unique x such that $G(\sigma\hat{\ }0)(x) \neq G(\sigma\hat{\ }1)(x)$ for any σ of length n and G^C be the path through G such that $G^C(d_n) = C(n)$. (As G is a 1-tree the x as required to define d_n is unique for each σ and the same for all of them.) These notions apply to finite 1-trees as G and finite binary strings as C with the obvious comment that there may only be finitely many d_n involved. If G is sufficiently generic for $\mathcal{P}_{1t}$ as required for Theorem 1treeofmin 9.6.10, and A is any set then it is clear that $A \leq_T G^A \vee G^{\bar{A}}$ as $n \in A$ if and only if $G^A(x) = 1$ where x is the nth place at which G^A and $G^{\bar{A}}$ differ (it is actually d_n). Thus we have two minimal degrees which join above $\mathbf{a}$. Our plan now is to take G_0 and G_1 two mutually sufficiently generic 1-trees for $\mathcal{P}_{1t}$ where the notion sufficiently generic now depends on A and assures that $(G_0^A \vee G_0^{\bar{A}}) \wedge (G_1^A \vee G_1^{\bar{A}})$.

Formally we consider the notion of forcing $\mathcal{P}_{2t}$ whose conditions consist of pairs $(P.Q)$ with each of P and Q a condition in P_{1t}. The ordering is given by $(\hat{P}.\hat{Q}) \leq_{\mathcal{P}_{2t}} (P.Q)$ if $\hat{P} \leq_{\mathcal{P}_{1t}} P$ and $\hat{Q} \leq_{\mathcal{P}_{1t}} Q$. In addition to the dense sets defined by requiring that each coordinate get into the dense sets from Theorem 1treeofmin 9.6.10 we have one more family of dense sets for the new meet requirement. For $(T, n) = P \in \mathcal{P}_{1t}$ we let P_n be the finite 1-tree given by restricting T to strings of length at most n. The argument is by now familiar. We let $A_e = \{(P.Q) | \exists x(\Phi_e^{(P_n^A \vee P_n^{\bar{A}})}(x) \downarrow \neq \Phi_e^{(Q_n^A \vee Q_n^{\bar{A}})}(x) \downarrow)$ or $(\forall (\hat{P}.\hat{Q}) \leq_{\mathcal{P}_{2t}} (P.Q))(\neg \exists x(\Phi_e^{(P_n^A \vee P_n^{\bar{A}})}(x) \downarrow \neq \Phi_e^{(Q_n^A \vee Q_n^{\bar{A}})}(x) \downarrow))\}$. Now if our generic meets A_e in condition $((P,n),(Q,m))$ and the first clause holds then clearly $\Phi_e^{(G_0^A \vee G_0^{\bar{A}})}(x) \downarrow \neq \Phi_e^{(G_1^A \vee G_1^{\bar{A}})}(x) \downarrow$ as, by the definition of extension in P_{2t}, $P_n^A \vee P_n^{\bar{A}}$ and $Q_n^A \vee Q_n^{\bar{A}}$ are initial segments of $G_0^A \vee G_0^{\bar{A}}$ and $G_1^A \vee G_1^{\bar{A}}$, respectively. On the other hand, if the second clause holds and $\Phi_e^{(G_0^A \vee G_0^{\bar{A}})}$ and $\Phi_e^{(G_1^A \vee G_1^{\bar{A}})}$ are total and equal, then we claim they are recursive in A. To compute $\Phi_e^{(G_0^A \vee G_0^{\bar{A}})}(x)$ find any finite extension R_m of P_n to a 1-tree of height m that is a subtree of P and such that $\Phi_e^{(R_n^A \vee R_n^{\bar{A}})}(x) \downarrow$. This R_m then extends to a full 1-tree R such that $(R,n) \leq_{\mathcal{P}_{1t}} (P,n)$. There must be one as $P_n \subseteq G_0$ and the computation of $\Phi_e^{(G_0^A \vee G_0^{\bar{A}})}(x)$ only requires finitely many levels of $G_0 \subseteq P$. If this were not the correct answer then, as for P and G_0, there would be a finite extension S_k of Q_n contained in Q which gives the same answer as $\Phi_e^{(G_1^A \vee G_1^{\bar{A}})}(x) \downarrow$. Again this S_k can be extended to an S such that $(S,n) \leq_{\mathcal{P}_{1t}} (Q,n)$. Then $((R,n),(S,n)) \leq_{\mathcal{P}_{2t}} (P,Q)$ but satisfies the first clause of A_e for the desired contradiction. ■

Exercise 9.6.14 *Show that the minimal degrees in* $\overline{\mathbf{GL}}_1$ *generate* $\mathcal{D}$.

Chapter 10

initialseg

Lattice Initial Segments of $\mathcal{D}$

Known results, history. Plan and goals here. Include all finite lattices and all countable distributive ones two of the major steps in previous process. new proof based on ... sufficient for all Applications. do two quantifier theory decidable and three undecidable.

First present the proof for recursive lattices which suffices for all our applications. Then indicate how to extend argument to cover all sublattices of any recursive lattice and so, for example, all distributive lattices.

10.1 Lattice Tables, trees and the notion of forcing

Our plan is to use lattice tables like those of latrep 6.3.4 to provide the basics of our embeddings of lattices as initial segments of $\mathcal{D}$. For simple embeddings in latembsec §6.3 we used a Cohen like forcing with conditions that were finite sequences of elements of our representation. In light of our move to trees in spectormin §9.2 to construct minimal degrees, it should not be surprising that we now move to conditions that are trees built on lattice tables Θ, i.e. maps $T : \Theta^{<\omega} \to \Theta^{<\omega}$ to provide the appropriate notions of forcing. The generic G that is built will then, as in latembsec §6.3, be an infinite sequence of elements from Θ. As a first approximation, the embedding will be given as before. For $x \in \mathcal{L}$, $x \longmapsto G_x$ where $G_x(n) = G(n)(x)$. (Recall that the elements of Θ are maps from $\mathcal{L}$ to $\mathbb{N}$.) Order, nonorder, join and meet are handled much as in latembsec §6.3. The key idea for making the embedding onto an initial segment will again be a type of e-splitting tree. While we want to deal with infinite lattices, a crucial component of the computation lemma for e-splitting trees (even in the minimal degree case) is that the trees are finitely branching. As long as they are finitely branching, one has a hope of determining the path taken by using Φ_e^G to choose among the e-splits. Thus we approximate our table by finite subsets Θ_i and consider trees T that at level i branch according to the elements of Θ_i. We will also have an associated decomposition of our given lattice $\mathcal{L} = \cup \mathcal{L}_i$. Now if one ignores the meet operation and the required interpolants it is easy to get a finite lattice table for a finite lattice. We call these upper semilattice (usl) tables. We postpone the meet interpolants for $\mathcal{L}_i$ to Θ_{i+1}. While this is not strictly necessary, it makes the construction of the tables much easier.

Moreover, we need a new type of interpolant to make the embeddings constructed be onto initial segments of $\mathcal{D}$ and we do not know if these could be incorporated as well into a finite lattice table for $\mathcal{L}_i$.

These new interpolants are called homogeneity interpolants. The idea here is that if we intend to force $\Phi_e^G \equiv_T G_x$ for some $x \in \mathcal{L}$ then using G_x we cannot distinguish among all the nodes σ in the tree at a given level n as many will have the same σ_x (be congruent modulo x). This suggests that we will want our trees to have some kind of homogeneity guaranteeing that what happens above one such σ is congruent to what happens above any other $\tau \equiv_x \sigma$. Of course, we need this property for every $x \in \mathcal{L}$.

With the above as a brief motivation, we now formally define the lattice tables that we use and the associated trees.

uslrepdef **Definition 10.1.1** *Let Θ be a set of maps from an usl $\mathcal{L}$ with least element 0 and greatest element 1 into $\mathbb{N}$. For $\alpha, \beta \in \Theta$ and $x \in \mathcal{L}$, we write $\alpha \equiv_x \beta$ (α is congruent to β modulo x) if $\alpha(x) = \beta(x)$. We write $\alpha \equiv_{x,y} \beta$ to indicate that α is congruent to β modulo both x and y. Such a Θ is an* usl table for $\mathcal{L}$ *if it contains the function that is 0 on every input (which we, by an abuse of notation, denote by 0) and for every $\alpha, \beta \in \Theta$ and $x, y, z \in \mathcal{L}$ the following properties hold:*

1. *$\alpha(0) = 0$.*
2. *(Differentiation) If $x \not\leq y$ then there are $\gamma, \delta \in \Theta$ such that $\gamma \equiv_y \delta$ but $\gamma \not\equiv_x \delta$.*
3. *(Order) If $x \leq y$ and $\alpha \equiv_y \beta$ then $\alpha \equiv_x \beta$.*
4. *(Join) If $x \vee y = z$ and $\alpha \equiv_{x,y} \beta$ then $\alpha \equiv_z \beta$.*

restriction **Notation 10.1.2** *If Θ is an usl representation for $\mathcal{L}$ and $\hat{\mathcal{L}} \subseteq \mathcal{L}$ then we denote the* restriction *of Θ to $\hat{\mathcal{L}}$ by $\Theta \restriction \hat{\mathcal{L}} = \{\alpha \restriction \hat{\mathcal{L}} | \alpha \in \Theta\}$. We also say that Θ is an* extension *of $\Theta \restriction \hat{\mathcal{L}}$. Note that as all our (upper or lower semi)lattices contain 1, the order property guarantees that if $\alpha \restriction \hat{\mathcal{L}} = \beta \restriction \hat{\mathcal{L}}$ then $\alpha = \beta$. Thus when we extend an usl representation $\hat{\Theta}$ for $\hat{\mathcal{L}}$ to one Θ for $\mathcal{L}$ (as in the constructions for Proposition extend 10.3.4 we can use the same $\alpha \in \hat{\Theta}$ to denote its unique extension in Θ.*

homo **Definition 10.1.3** *If Θ' and Θ are usl representations for $\mathcal{L}'$ and $\mathcal{L}$, respectively, $\hat{\mathcal{L}} \subseteq \mathcal{L}' \subseteq \mathcal{L}$ and $f : \Theta' \to \Theta$, then f is an $\hat{\mathcal{L}}$-*homomorphism *if, for all $\alpha, \beta \in \Theta'$ and $x \in \hat{\mathcal{L}}$, $\alpha \equiv_x \beta \Rightarrow f(\alpha) \equiv_x f(\beta)$.*

Theorem 10.1.4 (see Theorem repthm 10.3.1) *If $\mathcal{L}$ is a countable lattice, then there is an usl table Θ for $\mathcal{L}$ along with a nested sequence of finite sublower semilattices,* slsls, *$\mathcal{L}_i$ starting with $\mathcal{L}_0 = \{0, 1\}$ with union $\mathcal{L}$ and a nested sequence of finite subsets Θ_i with union Θ with both sequences recursive in $\mathcal{L}$ with the following properties:*

1. For each i, $\Theta_i \restriction \mathcal{L}_i$ is an usl table for $\mathcal{L}_i$.

2. There are meet interpolants for Θ_i in Θ_{i+1}, i.e. if $\alpha \equiv_z \beta$, $x \wedge y = z$ (with $\alpha, \beta \in \Theta_i$ and $x, y, z \in \mathcal{L}_i$) then there are $\gamma_0, \gamma_1, \gamma_2 \in \Theta_{i+1}$ such that $\alpha \equiv_x \gamma_0 \equiv_y \gamma_1 \equiv_x \gamma_2 \equiv_y \beta$.

3. For every sublowersemilattice $\hat{\mathcal{L}}$ of $\mathcal{L}_i$, $\hat{\mathcal{L}} \subseteq_{lsl} \mathcal{L}_i$, there are homogeneity interpolants for Θ_i with respect to $\hat{\mathcal{L}}$ in Θ_{i+1}, i.e. for every $\alpha_0, \alpha_1, \beta_0, \beta_1 \in \Theta_i$ such that $\forall w \in \hat{\mathcal{L}}(\alpha_0 \equiv_w \alpha_1 \to \beta_0 \equiv_w \beta_1)$, there are $\gamma_0, \gamma_1 \in \Theta_{i+1}$ and $\hat{\mathcal{L}}$-homomorphisms $f, g, h : \Theta_i \to \Theta_{i+1}$ such that $f : \alpha_0, \alpha_1 \mapsto \beta_0, \gamma_1$, $g : \alpha_0, \alpha_1 \mapsto \gamma_0, \gamma_1$ and $h : \alpha_0, \alpha_1 \mapsto \gamma_0, \beta_1$, i.e. $f(\alpha_0) = \beta_0$, $f(\alpha_1) = \gamma_1$ etc.

We prove this theorem in §lattablesec 10.3. Our goal in this and the next section is to prove that we have initial segment embeddings for all recursive lattices.

clatticeiso **Theorem 10.1.5** *Every recursive lattice $\mathcal{L}$ is isomorphic to an initial segment of $\mathcal{D}$.*

For the rest of this section and all of the next we fix a recursive lattice $\mathcal{L}$ and a sequence $\langle \mathcal{L}_i, \Theta_i \rangle$ for it as specified in Theorem repthm 10.3.1. We now move on to the definition of the trees that will be the conditions in our forcing relation.

latticetree **Definition 10.1.6** *A* tree T (for the sequence $\langle \mathcal{L}_i, \Theta_i \rangle$), *which we call simply a* tree *in this chapter ,is a recursive function such that for some $k \in \omega$ its domain is the empty string $\emptyset$ and all strings in the Cartesian product $\prod_{n=0}^{n=m} \Theta_{k+n}$ for each $m \in \omega$. We denote this number k by $k(T)$. For each $\sigma \in \operatorname{dom} T$, $T(\sigma) \in \prod_{n=0}^{n=q} \Theta_n$ for some $q \geq |\sigma| - 1$. Moreover, T has the following properties for all $\sigma, \tau \in \operatorname{dom} T$:*

1. *(Order) $\sigma \subseteq \tau \Rightarrow T(\sigma) \subseteq T(\tau)$.*

2. *(Nonorder) $\sigma | \tau \Rightarrow T(\sigma) | T(\tau)$. In fact, we specifically require that, for every $\sigma \in \prod_{n=0}^{n=m} \Theta_{k+n}$ and $\alpha \in \Theta_{k+m+1}$, $T(\sigma\hat{\ }\alpha) \supseteq T(\sigma)\hat{\ }\alpha$.*

3. *(Uniformity) For every fixed length l there is, for each $\alpha \in \Theta_{k+l}$, a string $\rho_{l,\alpha}$ so that, for a given l, all the $\rho_{l,\alpha}$ are of the same length independently of α and if $|\sigma| = l$ then $T(\sigma\hat{\ }\alpha) = T(\sigma)\hat{\ }\rho_{l,\alpha}$. Note that by the nonorder property (2), for fixed l and $\alpha \neq \beta$, $\rho_{l,\alpha} \neq \rho_{l,\beta}$, in fact, by our specific requirement, $\rho_{l,\alpha}(0) = \alpha$.*

Thus our trees T have branchings of width $|\Theta_{k(T)+n}|$ at level n and satisfy order and nonorder properties as for Spector forcing. In addition, they enjoy a strong uniformity property that will play a crucial role in our verifications.

subtree **Definition 10.1.7** *We say that a tree S is a* subtree *of a tree T, $S \subseteq T$, if $k(S) \geq k(T)$ and $(\forall \sigma \in \operatorname{dom} S)(\exists \tau \in \operatorname{dom} T)[S(\sigma) = T(\tau)]$.*

We note three useful facts that illuminate the structure of subtrees. The first says that the branchings on S follow those on T.

Lemma 10.1.8 *If S is a subtree of T then*
1. $S(\sigma) = T(\tau) \to (\forall \alpha \in \Theta_{k(S)+|\sigma|})(S(\sigma\hat{}\alpha) \supseteq T(\tau\hat{}\alpha)$
2. $\forall l \exists \rho_{l,\alpha} \forall \sigma, \tau(|\sigma| = l \ \&\ S(\sigma) = T(\tau) \ \to\ S(\sigma\hat{}\alpha) = T(\tau\hat{}\rho_{l,\alpha})$ for $\alpha \in \Theta_{k(S)+l}$ and
3. $[S] \subseteq [T]$.

Proof. The first fact follows immediately from our specific implementation of the nonorder property for trees (Definition latticetree 10.1.6(2)). The second follows from the uniformity requirements (3) of Definition latticetree 10.1.6 for S and T as well as property (2). the last is immediate from the definition. ■

Transitivity of the subtree relation should be clear but an even stronger claim is proven in Proposition trans 10.1.11. We mention some specific operations on trees that we will need later.

fulldef **Definition 10.1.9** *If T is a tree and $\sigma \in \operatorname{dom} T$ then $Fu(T,\sigma)$ or T_σ is defined by $T_\sigma(\tau) = T(\sigma\hat{}\tau)$. Clearly, $k(T_\sigma) = k(T) + |\sigma|$ and $T_\sigma \subseteq T$. Note that for $\sigma \in \operatorname{dom} T$ and $\tau \in \operatorname{dom} T_\sigma$, $(T_\sigma)_\tau = T_{\sigma\hat{}\tau}$. For a string $\mu \in \prod_{n=0}^{n=q} \Theta_n$ with $q \leq |T(\emptyset)| - 1$, we let T^μ (the* transfer tree *of T over μ) be the tree such that, for every $\sigma \in \operatorname{dom} T$, $T^\mu(\sigma)$ is the string gotten from $T(\sigma)$ by replacing its initial segment of length $q+1$ (which is contained in $T(\emptyset)$) by μ. We write T^μ_σ for $(T_\sigma)^\mu$. Finally, if T is a tree with $k(T) = k$ and $\sigma \in \operatorname{dom} T$ then we let $T^*_\sigma = T_\sigma \restriction \operatorname{dom} T$. Clearly, $k(T^*_\sigma) = k(T)$ and $T^*_\sigma \subseteq T$. Note that for $\sigma \in \operatorname{dom} T$ and $\tau \in \operatorname{dom} T^*_\sigma$, $(T^*_\sigma)^*_\tau = T^*_{\sigma\hat{}\tau}$.*

A crucial notion for our constructions is that of preserving the congruences of specified slsls of our given lattice $\mathcal{L}$.

prescong **Definition 10.1.10** *If $\hat{\mathcal{L}}$ is a finite slsl of $\mathcal{L}$ we say that a subtree S of T* preserves the congruences *of $\hat{\mathcal{L}}$, $S \subseteq_{\hat{\mathcal{L}}} T$, if $\hat{\mathcal{L}} \subseteq \mathcal{L}_{k(T)}$ and, whenever $x \in \hat{\mathcal{L}}$, $S(\sigma) = T(\tau)$, $\alpha \equiv_x \beta$, $S(\sigma\hat{}\alpha) = T(\tau\hat{}\mu)$ and $S(\sigma\hat{}\beta) = T(\tau\hat{}\nu)$, then $\mu \equiv_x \nu$. Here α and β are members of the appropriate Θ_i and μ and ν are sequences (necessarily of the same length m) of elements from the appropriate Θ_j's. We say that such sequences μ and ν are congruent modulo x, $\mu \equiv_x \nu$, if $\mu(j) \equiv_x \nu(j)$ for each $j < m$.*

trans **Proposition 10.1.11** *If $R \subseteq_{\mathcal{L}_1} S \subseteq_{\mathcal{L}_2} T$ and then $R \subseteq_{\mathcal{L}_1 \cap \mathcal{L}_2} T$.*

Proof. To see that $R \subseteq T$ note first that $k(R) \geq k(S) \geq k(T)$. Next suppose that $\rho \in \operatorname{dom} R$ and $\alpha \in \Theta_{k(R)+|\rho|}$. As $R \subseteq S$ we have a σ such that $R(\rho) = S(\sigma)$ and $R(\rho\hat{}\alpha) \supseteq S(\sigma\hat{}\alpha)$. As $S \subseteq T$ we have a τ such that $S(\sigma) = T(\tau)$ and $S(\sigma\hat{}\alpha) \supseteq T(\tau\hat{}\alpha)$. Thus $R(\rho) = T(\tau)$ and $R(\rho\hat{}\alpha) \supseteq T(\tau\hat{}\alpha)$ as required. As for the preservation of $\mathcal{L}_1 \cap \mathcal{L}_2$ congruences, suppose $R(\rho) = S(\sigma) = T(\tau)$, $x \in \mathcal{L}_1 \cap \mathcal{L}_2$, $\alpha_0, \alpha_1 \in \Theta_{k(R)+|\rho|}$ and $\alpha_0 \equiv_x \alpha_1$. Let $R(\rho\hat{}\alpha_i) = S(\sigma\hat{}\mu_i) = T(\tau\hat{}\nu_i)$. As $x \in \mathcal{L}_1$ and $R \subseteq_{\mathcal{L}_1} S$, $\mu_0 \equiv_x \mu_1$. As $x \in \mathcal{L}_2$ and

$S \subseteq_{\mathcal{L}_2} T$ it then follows by induction on the (by uniformity, necessarily common) length of μ_i that $\nu_0 \equiv_x \nu_1$ as required.

The details of this induction follow. Write $\nu_i = \nu_i^0\hat{\ }\cdots\hat{\ }\nu_i^s$ where $S(\sigma\hat{\ }\mu_i(0)\cdots\hat{\ }\mu_i(t)) = T(\tau\hat{\ }\nu_i^{0\hat{\ }\cdots\hat{\ }}\nu_i^t)$. Then inductively $\mu_0(j) \equiv_x \mu_1(j)$ gives $\nu_0^j \equiv_x \nu_1^j$. For $j = 0$ this follows directly from Definition prescong 10.1.10. For the inductive step, consider, without loss of generality, the case $j = 1$. We have $S(\sigma\hat{\ }\mu_0(0)\hat{\ }\mu_0(1)) = T(\tau\hat{\ }\nu_0^0\hat{\ }\nu_0^1)$ and $S(\sigma\hat{\ }\mu_1(0)\hat{\ }\mu_1(1)) = T(\tau\hat{\ }\nu_1^0\hat{\ }\nu_1^1)$. Consider $S(\sigma\hat{\ }\mu_0(0)\hat{\ }\mu_1(1)) = T(\tau\hat{\ }\nu_0^0\hat{\ }\nu)$ for some ν. By the uniformity clause (3) of Definition latticetree 10.1.6, there is a ζ such that $S(\sigma\hat{\ }\mu_0(0)\hat{\ }\mu_1(1)) = S(\sigma\hat{\ }\mu_0(0))\hat{\ }\zeta$ and $S(\sigma\hat{\ }\mu_1(0)\hat{\ }\mu_1(1)) = S(\sigma\hat{\ }\mu_1(0))\hat{\ }\zeta$. Thus $T(\tau\hat{\ }\nu_0^0\hat{\ }\nu) = T(\tau\hat{\ }\nu_0^0)\hat{\ }\zeta$ and $T(\tau\hat{\ }\nu_1^0\hat{\ }\nu_1^1) = T(\tau\hat{\ }\nu_1^0)\hat{\ }\zeta$. Again, by the uniformity clause and the uniqueness of the $\rho_{l,\alpha}$ there (iterated $|\nu|$ times), $\nu = \nu_1^1$. Finally, by Definition prescong 10.1.10 again, $\nu \equiv_x \nu_0^1$ as $\mu_1(1) \equiv_x \mu_0(1)$ and so $\nu_1^1 \equiv_x \nu_0^1$ as required. ■

We now present the notion of forcing for constructing our embedding of $\mathcal{L}$ as an initial segment of $\mathcal{D}$.

defforcing **Definition 10.1.12** *The* forcing conditions *P our notion of forcing $\mathcal{P}$ are trees T (for $\langle \mathcal{L}_i, \Theta_i \rangle$). We say $T_1 \leq_{\mathcal{P}} T_0$, if $T_1 \subseteq_{K(T_0)} T_0$ where, as often, we denote $\mathcal{L}_{k(T)}$ by $K(T)$. We let $V(T) = T(\emptyset)$. The top element of $\mathcal{P}$ consists of the identity tree Id (which has $k(Id) = 0$).*

fulllem **Lemma 10.1.13** *If T is a tree, $\sigma \in \operatorname{dom} T$ and $\hat{\mathcal{L}} \subseteq \mathcal{L}_{k(T)}$, then $T_\sigma \subseteq_{\hat{\mathcal{L}}} T$. If $\sigma, \tau \in \operatorname{dom} T$ are of the same length and $S \leq_{\mathcal{P}} T_\sigma$ then $S^{T(\tau)} \leq_{\mathcal{P}} T_\tau$. We also have that $T_\sigma^{T(\tau)} = T_\tau$.*

Proof. The first assertions follow directly from the definitions. The last two follow from the uniformity assumption on our trees. ■

It is easy to see that sets $C_n = \{P |\ |V(P)| > n \ \&\ k(P) > n\}$ are dense. Just extend to some P_σ. We assume that any generic filter $\mathcal{G}$ we consider meets these sets. It then determines a generic function $G \in \prod_{n=0}^{\infty} \Theta_n$, i.e. a function on ω with $G(n) \in \Theta_n$. On this basis we could naively try to define our embedding of $\mathcal{K}$ into $\mathcal{D}$ as we did in latembsec §6.3: For $x \in \mathcal{K} \subseteq \mathcal{L}$ we let $G_x : \omega \to \omega$ be defined by $G_x(n) = G(n)(x)$. The desired image of x would then be $\deg(G_x)$. Now the order and join properties of usl representations guarantee that this embedding preserves order and join (on all of $\mathcal{L}$ even). If $x \leq y$ then by the order property we can (recursively in the table $\langle \Theta_i \rangle$) calculate $G_x(m)$ from $G_y(m)$ by finding any $\alpha \in \Theta_m$ with $\alpha(y) = G_y(m)$ and declaring that $G_x(m) = \alpha(x)$. (Such an α exists since $G(m)$ is one.) Similarly if $x \vee y = z$ then, by the join property, we can calculate $G_z(m)$ from $G_x(m)$ and $G_y(m)$ by finding any $\alpha \in \Theta_m$ such that $\alpha(x) = G_x(m)$ and $\alpha(y) = G_y(m)$ and declaring that $G_z(m) = \alpha(z)$. (Again $G(m)$ is such an α.)

Were congruences modulo x always preserved for every x, we could directly carry out the diagonalization and other requirements as well for this definition of G_x. In actuality, however, not all congruences are preserved as we refine to various subtrees in our construction. Thus we must modify the definition of the images in $\mathcal{D}$ and provide nice representations of the degree corresponding to x.

Definition 10.1.14 *If $\mathcal{G}$ is a generic filter meeting the dense sets C_n, G the corresponding generic element of $\prod_{n=0}^{\infty} \Theta_n$, $P \in \mathcal{G}$ and $x \in K(P)$ then G^P is the sequence $\langle \alpha_n | n \in \omega \rangle$ where $P(\langle \alpha_n | n < m \rangle) \subseteq G$ for every m. (Thus $\langle \alpha_n \rangle$ is the path that G follows in the domain of P. In particular, $G = G^{Id}$. It is obvious from the definitions that G is a path on (i.e. in the range of) Q for every $Q \in \mathcal{G}$.) We define $G^P_x(n)$ as $\alpha_n(x)$.*

The crucial point is that the degree of G^P_x does not depend on P once $x \in K(P)$.

Lemma 10.1.15 *If $x \in K(P), K(Q)$ for P, Q in a generic $\mathcal{G}$, then $G^P_x \equiv_T G^Q_x$.*

Proof. As there is an $R \leq_{\mathcal{P}} P, Q$ in $\mathcal{G}$ by the compatibility of all conditions in a generic filter, it suffices to consider the case that $Q \leq_{\mathcal{P}} P$. Let $G^P = \langle \alpha_n \rangle$ and $G^Q = \langle \beta_n \rangle$. By the definition of subtree there is for each n an $m(n)$ such that $Q(\langle \beta_s | s < n \rangle) = P(\langle \alpha_s | s < m(n) \rangle)$ and we can compute the function m recursively in the trees. (By the uniformity of the trees, there is, for each n, a unique $m(n)$ such that $|Q(\sigma)| = |P(\tau)|$ for every σ of length n and every τ of length $m(n)$.) Moreover, by our definition of subtree, $\beta_n = \alpha_{m(n)}$. Thus $G^Q_x(n) = \beta_n(x) = \alpha_{m(n)}(x) = G^P_x(m(n))$ and so $G^Q_x \leq_h G^P_x$. The other direction depends on the congruence preservations for x implied by $Q \subseteq_{K(P)} P$.

Suppose that we have, by recursion, determined $G^P_x(i) = \alpha_i(x)$ for $i \leq m(n)$. The next step followed by G in Q is $\beta_{n+1} = \alpha_{m(n)+1}$. It corresponds to the sequence $\langle \alpha_i | m(n) + 1 \leq i < m(n+1) \rangle$. The definition of $\subseteq_{K(P)}$ implies that $\langle \alpha_i(x) | m(n) + 1 \leq i < m(n+1) \rangle$ is uniquely determined by $\beta_{n+1}(x)$ to continue the recursion. ■

Thus given a generic $\mathcal{G}$ we can define a map from $\mathcal{L}$ into $\mathcal{D}$ by sending $x \in \mathcal{L}$ to deg(G^P_x) for any $P \in \mathcal{G}$ with $x \in K(P)$. Our proof plans above as in latembsec §6.3 for the preservation of order and join now work here as well simply by applying them to G^P on P (in place of G on Id) for any $P \in \mathcal{G}$ with $x, y, z \in K(P)$. Thus we only need to verify the preservation of nonorder and that our map is onto an initial segment of $\mathcal{D}$. (Note that meet is preserved once we know that the mapping is an order isomorphism of the lattice $\mathcal{L}$ onto an initial segment of $\mathcal{D}$ as meet is definable from order. Of course, this argument would apply to join as well but no new work is needed to note that join is preserved. It is also worth commenting that we use the join structure in the usl tables as well as the meet interpolants in the proof that the embedding is onto an initial segment of $\mathcal{D}$.)

10.2 Initial segment conditions

To assure that our embedding preserves nonorder we want to show, for any $x \not\leq y$ in $\mathcal{K}$, condition P with $x, y \in K(P)$ and Φ_e, that there is a $Q \leq_{\mathcal{P}} P$ such that for any $G \in [Q]$ $\Phi_e^{G^P_y} \neq G^P_x$ and a $Q \leq_{\mathcal{P}} P$ and $x \in K(Q)$ such that for any $G \in [Q]$ for which Φ^G_e is total, $\Phi^G_e \equiv_T G^Q_x$. These two results would then finish the proof of our theorem. We begin with the analog of total subtrees of Definition totdef 9.2.17 and the corresponding dense sets that make our task simpler.

totldef **Definition 10.2.1** *Let T be a condition in $\mathcal{P}$. If for every $\sigma \in \operatorname{dom} T$ and every x there is a $\tau \supseteq \sigma$ such that $\Phi_e^{T(\tau)}(x)\downarrow$ then we define a subtree $S = Totl(T, e)$ with the same domain by recursion on the length of $\sigma \in \operatorname{dom} T$. We begin with $S(\emptyset) = T(\emptyset)$. Suppose for every $\sigma_i \in \operatorname{dom} T$ of length n there is a τ_i such that we have defined $S(\sigma_i) = T(\tau_i)$ for $i < m$. We now list the α such that we must define $S(\rho_i\hat{}\alpha)$ to get the next level of S as $\langle \alpha_j | j < s\rangle$. We proceed to define ρ_l for each $l = \langle i, j\rangle$ with $i < m$ and $j < s$. (For convenience we assume these are the $l < r = m \cdot s$.) For $l = 0 = \langle 0, 0\rangle$ we search for the first $\rho \supseteq$ such that $\Phi_e^{T(\tau_0\hat{}\alpha_0\hat{}\rho)}(|\sigma|)\downarrow$. One exists by our assumption. We then set $\rho = \rho_0$. If we have defined ρ_l for $l \leq q$ and $\mu_q = \rho_0\hat{}\ldots\hat{}\rho_q$ then we let ρ_{q+1} where $q + 1 = \langle \hat{\imath}, \hat{\jmath}\rangle$ be the first ρ such that $\Phi_e^{T(\tau_{\hat{\imath}}\hat{}\alpha_{\hat{\jmath}}\hat{}\mu_q\hat{}\rho)}(|\sigma|)\downarrow$. We now let μ be the concatenation of the ρ_l for $l < r$ and set $S(\sigma_i\hat{}\alpha_j) = T(\tau_i\hat{}\alpha_j\hat{}\mu)$.*

Definition 10.2.2 *If T is a tree and $(\forall\sigma)(\forall x < |\sigma|)(\Phi_e^{T(\sigma)}(x)\downarrow$, we say that T is e-*total *and we denote $\Phi_e^{T(\sigma)}(n)$ for $n < |\sigma|$ by $q_T(n, \sigma)$.*

Lemma 10.2.3 *If T and $Totl(T) = S$ is defined then $S \leq_{\mathcal{P}} T$ and S is e-*total.

Proof. The second claim is immediate from the definition of $Totl(T)$. As for the first, it is immediate that $\operatorname{dom} S = \operatorname{dom} T$ and that, since T is a tree, that S is a subtree of T. As the definition of S has $S(\sigma_i\hat{}\alpha_j) = T(\tau_i\hat{}\alpha_j\hat{}\mu)$ for a single μ over all the nodes $\sigma_i\hat{}\alpha_j$ of level $n + 1$, it is also clear that S preserves all the congruences of $K(T)$. ■

totl **Lemma 10.2.4** *The sets $B_e = \{P | \exists x \forall \sigma(\Phi_e^{P(\sigma)}(x)\uparrow)$ or P is e-*total$\}$ *are dense in $\mathcal{P}$.*

Proof. Suppose we are given P and e. If there is an x and a σ such that $\Phi_e^{P(\tau)}(x)\uparrow$ for every $\tau \supseteq \sigma$, then clearly P_σ (or P_σ^*) satisfies the first clause in the definition of B_e. Otherwise, $Totl(P, e)$ satisfies the second clause by the last Lemma. ■

diag **Proposition 10.2.5 (Diagonalization)** *For any $x \not\leq y$ in $\mathcal{L}$, $e \in \mathbb{N}$ and condition P with $x, y \in K(P)$, there is an $Q \leq P$ such that $\forall G \in [Q]$, if Φ_e^G is total then $\Phi_e^{G_y^P} \neq G_x^P$.*

Proof. There is clearly an j such that $\Phi_e^{G_y^P} = \Phi_j^G$. By Lemma totl 10.2.4 we may assume that P is j-total. We then choose any $\alpha_0, \alpha_1 \in \Theta_{k(P)}$ such that $\alpha_0 \equiv_y \alpha_1$ but $\alpha_0 \not\equiv_x \alpha_1$. Such α_0 and α_1 exist by the differentiation property of usl tables. Let $\beta_i = (\alpha_i)^{|\sigma|}$, i.e. the concatenation of $|\sigma|$ many copies of α_i for $i \in \{0, 1\}$. Consider then the conditions P_{β_i} and any $G_i \in [P_{\beta_i}]$. Of course, $(G_i)_x^P(|\sigma|) = \alpha_i(x)$ while $\Phi_i^{P_{\beta_i}(\emptyset)}(|\sigma|)\downarrow$ for $i = 0, 1$ by Lemma totl 10.2.4. As the β_i, for $i = 0, 1$, are congruent modulo y and $y \in K(P)$, the initial segments of G_y^P that $P_{\beta_i}(\emptyset)$ determine are equal. Thus the $\Phi_i^{P_{\beta_i}(\emptyset)}(|\sigma|) = \Phi_e^{G_y^P}(|\sigma|)$ are convergent and equal. So for one $i \in \{0, 1\}$, $\Phi_i^{P_{\beta_i}(\emptyset)}(|\sigma|) \neq \alpha_i(x)$. For that i, P_{β_i} is the Q required in the Lemma. ■

We turn now to the requirement that the image of $\mathcal{K}$ under our embedding form an initial segment of $\mathcal{D}$. This argument is somewhat more complicated than those above and uses both the meet and homogeneity interpolants.

We begin with the notion of an e-splitting appropriate to our trees and a lemma about such splittings.

defsplit **Definition 10.2.6** *Given a Φ_e and an e-total tree Q. we say that σ and τ with $|\sigma| = |\tau|$ are an e-splitting or e-split on Q (modulo w) if ($\sigma \equiv_w \tau$ and) there is an $n < |\sigma|$ such that $q_T(n,\sigma) \neq q_T(n,\tau)$. If $R \leq Q, R(\mu) = Q(\sigma), R(\nu) = Q(\tau)$ and σ and τ e-split (modulo w) on Q then we also say that μ and ν e-split on R (modulo w).*

meetsplit **Lemma 10.2.7** *Given an e-total condition Q, there is a $\rho \in \operatorname{dom} Q$ such that the set $Sp(\rho) = \{w \in K(Q)|$ there are no σ, τ that e-split on Q^*_ρ modulo $w\}$ is maximal. Moreover, this maximal set is closed under meet and so has a least element z.*

Proof. Let $k = k(Q)$ and $\hat{\mathcal{K}} = K(Q)$. As $\hat{\mathcal{K}}$ is finite there is clearly a ρ such that $Sp(\rho)$ is maximal. Note that then $Sp(\mu) = Sp(\rho)$ for any $\mu \supseteq \rho$ with $\mu\backslash\rho \in \operatorname{dom} Q^*_\rho$ as $Q^*_\mu \subseteq_{\hat{\mathcal{K}}} Q^*_\rho$. Consider any $x, y \in Sp(\rho)$ with $x \wedge y = w$. As $\hat{\mathcal{K}} \subseteq_{lsl} \mathcal{L}$, $w \in \hat{\mathcal{K}}$. To show that $Sp(\rho)$ is closed under meet it suffices (by the maximality of $Sp(\rho)$) to show that there is no e-splitting on $Q^*_{\rho\hat{}0}$ modulo w. Remember that, by definition, $k = k(Q^*_{\rho\hat{}0}) = k(Q^*_\rho) = k(Q)$. Suppose there were such a split $\bar{\mu}$ and $\bar{\nu}$, each of length m. By our definition of $Q^*_{\rho\hat{}0}$, $\bar{\mu}, \bar{\nu} \in \prod_{n=0}^{n=m} \Theta_{k+n}$. In Q^*_ρ at the corresponding levels, however, there are branchings for all elements of Θ_{k+n+1}. (That is there are, for example, successors of $Q^*_\rho(0\hat{}\mu \restriction n+1)$ for every element of Θ_{k+n+1} while in $Q^*_\rho(\mu \restriction n+1)$ there are ones only for the elements of Θ_{k+n}.) Thus, by the existence of meet interpolants for Θ_{k+n} in Θ_{k+n+1}, there are $\bar{\gamma}_0, \bar{\gamma}_1, \bar{\gamma}_2 \in \prod_{n=0}^{n=m} \Theta_{k+n+1}$ such that for each $j \leq m$, the $\bar{\gamma}_i(j)$ for $i \in \{0,1,2\}$ are meet interpolants for $\bar{\mu}(j)$ and $\bar{\nu}(j)$, i.e. $\bar{\mu} \equiv_x \bar{\gamma}_0 \equiv_y \bar{\gamma}_1 \equiv_x \bar{\gamma}_2 \equiv_y \bar{\nu}$. As $\bar{\mu}$ and $\bar{\nu}$ form a e-splitting on $Q^*_{\rho\hat{}0}$ so do one of the successive pairs such as $0\hat{}\bar{\gamma}_0$, $0\hat{}\bar{\gamma}_1$ on Q^*_ρ. But then $0\hat{}\bar{\gamma}_0$ and $0\hat{}\bar{\gamma}_1$ would be an e-split on Q^*_ρ congruent modulo y for a contradiction. (The situations for the other pairs are the same but perhaps with x in place of y.) ■

We now build the analog of the e-splitting subtrees of Definition esplittree 9.2.12.

splittree **Proposition 10.2.8** *Given an e-total Q with $k(Q) = k$ and $K(Q) = \hat{\mathcal{K}}$ with ρ and z as in Lemma meetsplit 10.2.7, there is a condition $S \leq Q^*_\rho$ with $k(S) = k$ such that any $\sigma, \tau \in$ $\operatorname{dom} S(= \operatorname{dom} Q)$ with $\sigma \not\equiv_z \tau$ e-split on S. (Of course, by the choice of ρ and z there are no e-splits on Q^*_ρ which are congruent modulo z.) Such a tree S is called a $z-e$-*splitting *tree.*

Proof. We define $S(\sigma)$ (with $k(S) = k$) by induction on $|\sigma|$ beginning, of course, with $S(\emptyset) = Q^*_\rho(\emptyset)$. Suppose we have defined $S(\sigma) = Q^*_\rho(\tau_\sigma)$ for all σ of length n. We must define $S(\sigma\hat{}\alpha)$ for all such σ and appropriate α as extensions $Q^*_\rho(\tau_{\sigma\hat{}\alpha})$ of $Q^*_\rho(\tau_\sigma\hat{}\alpha)$

obeying all the congruences in $\hat{\mathcal{K}}$, i.e. if $x \in \hat{\mathcal{K}}$ and $\alpha \equiv_x \beta$ then $\tau_{\sigma\hat{}\alpha} \equiv_x \tau_{\sigma\hat{}\beta}$. We list the σ of length $n+1$ as $\sigma_i\hat{}\alpha_i$ for $i < m = |\prod_{j=0}^{j=n} \Theta_{k+j}|$ and define by induction on $r < l = m(m+1)/2$ (the number of pairs $\{i,j\}$ with $i,j < m$) strings $\rho_{i,r}$ simultaneously for all $i < m$. At the end of our induction we will set $\tau_{\sigma_i\hat{}\alpha_i} = \tau_{\sigma_i}\hat{}\alpha_i\hat{}\rho_{i,0}\hat{}\ldots\hat{}\rho_{i,l-1}$. For this to succeed it suffices to maintain uniformity and guarantee, for every $i,j < m$ and $w \in \hat{\mathcal{K}}$, that $\alpha_i \equiv_w \alpha_j \Rightarrow \rho_{i,r} \equiv_w \rho_{j,r}$ for every $r < l$ and that if $\alpha_i \not\equiv_z \alpha_j$ then $\tau_{\sigma_i}\hat{}\alpha_i\hat{}\rho_{i,0}\hat{}\ldots\hat{}\rho_{i,r}$ and $\tau_{\sigma_j}\hat{}\alpha_j\hat{}\rho_{j,0}\hat{}\ldots\hat{}\rho_{j,r}$ e-split on Q^*_ρ where $r < l$ is (the code for) $\{i,j\}$.

By induction on $r < l$ we suppose we have $\tau_{\sigma_i}\hat{}\rho_{i,0}\hat{}\ldots\hat{}\rho_{i,r-1} = \nu_i$ for all $i < m$ and that $\{p,q\}$ is pair number r. If $\alpha_p \equiv_z \alpha_q$ there is no requirement to satisfy and we let $\rho_{i,r} = \emptyset$ for every i. Otherwise, let w be the largest $y \in \mathcal{L}_{k+n}$ such that $\alpha_p \equiv_y \alpha_q$. (To see that there is a largest such y, first note that $\mathcal{L}_{k+n}$ is a lattice as it is a finite lsl. As Θ_{k+n} is an usl table for $\mathcal{L}_{k+n}$, if $\alpha_p \equiv_{u,v} \alpha_q$ for $u,v \in \mathcal{L}_{k+n}$ then $\alpha_p \equiv_t \alpha_q$ where t is the least element of $\mathcal{L}_{k+n}$ above both u and v (their join from the viewpoint of $\mathcal{L}_{k+n}$). Thus, there is a largest y as desired.) Of course, $z \not\leq w$. By our choice of z there are $\sigma, \tau \in \prod_{t=0}^{t=c} \Theta_{k+t}$ such that ν_p extended by σ and τ form an e-splitting congruent modulo w on Q^*_ρ. (We can find such a split on $Q^*_{\nu_p}$ by the definition of ρ and z and our assumption on w. It translates into such σ and τ.) Consider $\nu_q\hat{}\tau$. It must form an e-splitting on Q^*_ρ with one of $\nu_p\hat{}\sigma$ and $\nu_p\hat{}\tau$ by the basic properties of Q. If it splits with the latter string then we can set $\rho_{i,r+1} = \tau$ and clearly fulfill the requirements for this pair $\{p,q\}$ both for congruence modulo w (as all new extensions are identical) and e-splitting. Of course, uniformity is maintained as the $\rho_{i,r+1}$ are the same for all i. Thus we assume that $\nu_p\hat{}\sigma$ and $\nu_q\hat{}\tau$ e-split on Q^*_ρ. We now use our homogeneity interpolants.

We know that w is the largest $y \in \mathcal{L}_{n+k}$ such that $\alpha_p \equiv_y \alpha_q$ and that $\sigma \equiv_w \tau$. Thus for any $x \in \hat{\mathcal{K}} \subseteq \mathcal{L}_{k+n}$ if $\alpha_p \equiv_x \alpha_q$ then $x \leq w$ and so $\sigma \equiv_x \tau$. By Theorem repthm 10.3.1(3) we can now find homogeneity interpolants $\gamma_0(s), \gamma_1(s)$ in Θ_{k+s+1} and associated $\hat{\mathcal{K}}$-homomorphisms $f_s, g_s, h_s : \Theta_{k+s} \to \Theta_{k+s+1}$ such that $f_s : \alpha_p, \alpha_q \mapsto \sigma(s), \gamma_1(s)$, $g_s : \alpha_p, \alpha_q \mapsto \gamma_0(s), \gamma_1(s)$ and $h_s : \alpha_p, \alpha_q \mapsto \gamma_0(s), \tau(s)$ for each $s < |\sigma| = |\tau|$. (We let $\alpha_0 = \alpha_p$, $\alpha_1 = \alpha_q$, $\beta_0 = \sigma(s)$, $\beta_1 = \tau(s)$, $\hat{\mathcal{L}} = \hat{\mathcal{K}}$ and $i = k+s$ in the Theorem.) Note that the branchings in Q^*_ρ are at some levels up from the corresponding ones in $Q^*_{\nu_p}$ or $Q^*_{\nu_q}$ on which we chose σ and τ. Thus these homogeneity interpolants are available within the branchings in Q^*_ρ. As $\nu_p\hat{}\sigma$ and $\nu_q\hat{}\tau$ e-split on Q^*_ρ one of the pairs $\nu_p\hat{}\sigma, \nu_q\hat{}\bar{\gamma}_1$; $\nu_p\hat{}\bar{\gamma}_0, \nu_q\hat{}\bar{\gamma}_1$ and $\nu_p\hat{}\bar{\gamma}_0, \nu_q\hat{}\tau$ must also e-split on Q^*_ρ. Suppose, for the sake of definiteness, it is the second pair $\nu_p\hat{}\bar{\gamma}_0, \nu_q\hat{}\bar{\gamma}_1$. In this case, we let $\rho_{i,r+1}(s) = g_s(\alpha_i)$ for every i and s. Note that uniformity is maintained as $\rho_{i,r+1}(s)$ depends only on α_i. We use f_s or h_s in place of g_s if the e-splitting pairs are $\nu_p\hat{}\sigma, \nu_q\hat{}\bar{\gamma}_1$ or $\nu_p\hat{}\bar{\gamma}_0, \nu_q\hat{}\tau$, respectively. By the homomorphism properties of the interpolants these extensions preserve all the congruences in $\hat{\mathcal{K}}$ between any α_i and α_j as required to complete the induction and our construction of an e-splitting tree . ■

We now conclude the proof that our embedding maps onto an initial segment of $\mathcal{D}$. by showing that for $G \in [S]$ with S a $z - e$-splitting tree, $\Phi_e^G \equiv_T G_z^S$. The proof is analogous to that of the Computation Lemma ($\overset{\texttt{complemma}}{9.2.14}$).

comp **Lemma 10.2.9** *If S is a $z - e$-splitting tree and $G \in [S]$ then $\Phi_e^G \equiv_T G_z^S$.*

Proof. We first show that $\Phi_e^G \leq_T G_z^S$. Consider any n. Using G_z^S we can find all the $\sigma \in \operatorname{dom} S$ of length n such that $\sigma(l) = G_z^S(l)$ for every $l \leq n$. All of these σ are congruent modulo z and so all S_σ force the same value for Φ_e^G at n. As $S(\sigma)$ is an initial segment of G for one of these σ, this value must be $\Phi_e^G(n)$. We next argue that $G_z^S \leq_T \Phi_e^G$. Consider all $\sigma, \tau \in \operatorname{dom} S$ of length n. If $\sigma \not\equiv_z \tau$ then S_σ and S_τ force different values for Φ_e^G at some $l < n$. Thus using $\Phi_e^G \upharpoonright n$ we can find the unique congruence class modulo z consisting of those σ such that $S(\sigma)$ is not ruled out as a possible initial segment of G. For one σ in this class, $S(\sigma)$ is an initial segment of G and as all the σ in this class are congruent modulo z, they all determine the same values of $G_z^S \upharpoonright n$ which must then be the correct value. ■

We have now completed the proof that any generic filter $\mathcal{G}$ (deciding all sentences and meeting the dense sets provided by Lemma $\overset{\texttt{totl}}{10.2.4}$ and Propositions $\overset{\texttt{diag}}{10.2.5}$ and $\overset{\texttt{splittree}}{10.2.8}$) provides an embedding of $\mathcal{L}$ onto an initial segment of $\mathcal{D}$ that sends x to $\deg(G_x^P)$ (for any $P \in \mathcal{G}$ with $x \in K(P)$). This establishes Theorem $\overset{\texttt{reclatticeiso}}{10.1.5}$ given our lattice table theorem whose proof we provide in §$\overset{\texttt{lattablesec}}{10.3}$. We now indicate how to modify Theorem $\overset{\texttt{reclatiso}}{\textbf{??}}$ so as to apply to any sublattice $\mathcal{K}$ of a recursive lattice.

clatticeiso **Theorem 10.2.10** *If $\mathcal{K}$ is a sublattice of a recursive lattice $\mathcal{L}$ then $\mathcal{K}$ is isomorphic to an initial segment of $\mathcal{D}$.*

Proof. The changes needed to the proof of Theorem $\overset{\texttt{reclatticeiso}}{10.1.5}$ are mostly notational. The forcing conditions are now pairs $\langle T, \hat{\mathcal{K}} \rangle$ where T is a tree (for $\langle \mathcal{L}_i, \Theta_i \rangle$) and $\hat{\mathcal{K}}$ is a finite slsl of $\mathcal{K} \cap \mathcal{L}_{k(T)}$. We say that $\langle T_1, \mathcal{K}_1 \rangle \leq_{\mathcal{P}} \langle T_0, \mathcal{K}_0 \rangle$ if $T_1 \subseteq_{\mathcal{K}_0} T_0$ and $\mathcal{K}_1 \supseteq \mathcal{K}_0$. We let $V(\langle T, \hat{\mathcal{K}} \rangle) = T(\emptyset)$. If $P = \langle T, \hat{\mathcal{K}} \rangle$ is a condition we let $K(P) = \hat{\mathcal{K}}$, $Tr(P) = T$ and $k(P) = k(T)$. In following much of the original proof, one should often simply replace a condition P by $Tr(P)$ when $K(P)$ is fixed. Along these lines, for example, we use P_σ, P_σ^*, P^τ and P_σ^τ to stand for $\langle Tr(P)_\sigma, K(P) \rangle$, $\langle Tr(P)_\sigma^*, K(P) \rangle$, $\langle Tr(P)^\tau, K(P) \rangle$ and $\langle Tr(P)_\sigma^\tau, K(P) \rangle$, respectively. The top element of $\mathcal{P}$ consists of the identity tree Id (which has $k(Id) = 0$) and the slsl $\mathcal{L}_0 = \{0, 1\}$.

The basic dense sets C_n that we assume are met by any generic are now extended to include, for each $x \in \mathcal{K}$ the sets $\{P | x \in K(P)\}$. to see that these are dense, consider any $Q \in \mathcal{P}$ and $x \in \mathcal{K}$. Let $\mathcal{K}'$ be the slsl of $\mathcal{K}$ generated by $K(Q)$ and x and let $i \geq k(Q)$ be such that $\mathcal{K}' \subseteq \mathcal{L}_i$. Define S with $k(S) = i$ by $S(\sigma) = Tr(Q)(0^{i-k(Q)}{}^\frown\sigma)$. Clearly, $\langle S, \mathcal{K}' \rangle \leq_{\mathcal{P}} Q$ and is in the required set.

The definition of the embedding, now from $\mathcal{K}$, into $\mathcal{D}$ is the same as before noting that $K(P)$ is now the second coordinate of P rather than simply $\mathcal{L}_{k(P)}$. The operations on trees and proofs used to verify the diagonalization (for $x, y \in \mathcal{K}$) and initial segment properties (with $\Phi_e^G = G_z^P$ for $z \in \mathcal{K}(\mathcal{Q}) \subseteq \mathcal{K}$) are now essentially the same. Just keep in mind that they are applied to $Tr(P)$ and $K(P)$ does not change. ■

This version of the theorem provides initial segment embeddings for many nonrecursive lattices. As an example we have the following corollary.

Corollary 10.2.11 *Every countable distributive lattice is isomorphic to an initial segment of $\mathcal{D}$.*

Proof. There is a recursive universal countable distributive lattice. In fact, every countable distributive lattice can be embedded into the atomless Boolean algebra.?? ■

Exercise 10.2.12 *Prove that the embedding of our recursive lattice $\mathcal{L}$ can be taken to be into $\mathcal{D}(\leq\mathbf{0}'')$ and, indeed that the generic G constructed has double jump $0''$. For embeddings of a sublattice $\mathcal{K}$ of $\mathcal{L}$ determine where the embedding lies and what can be said about G''.*

Exercise 10.2.13 *Prove that the embedding of our recursive lattice $\mathcal{L}$ is onto an initial segment of both the tt and wtt degrees. (Hint: recall Exercise* Odomtt *8.1.2.)*

Exercise 10.2.14 *Theorem* reclatticeiso *10.1.5 relativizes to any degree $\mathbf{a}$ and so every countable lattice $\mathcal{L}$ (with 0 and 1) is isomorphic to a segment of $\mathcal{D}$, i.e. to $[\mathbf{a},\mathbf{b}] = \{\mathbf{x} | \mathbf{a} \leq \mathbf{x} \leq \mathbf{b}\}$ for some $\mathbf{b}$ where $\mathbf{a}$ is the degree of $\mathcal{L}$. Indeed, we may take $\mathbf{b}'' = \mathbf{a}''$.*

lattablesec

10.3 Constructing lattice tables

repthm

Theorem 10.3.1 *If $\mathcal{L}$ is a countable lattice then there is an usl table Θ of $\mathcal{L}$ along with a nested sequence of finite slsls $\mathcal{L}_i$ starting with $\mathcal{L}_0 = \{0,1\}$ with union $\mathcal{L}$ and a nested sequence of finite subsets Θ_i with union Θ with both sequences recursive in $\mathcal{L}$ with the following properties:*

1. *For each i, $\Theta_i \upharpoonright \mathcal{L}_i$ is an usl table of $\mathcal{L}_i$.*

2. *There are* meet interpolants *for Θ_i in Θ_{i+1}, i.e. if $\alpha \equiv_z \beta$, $x \wedge y = z$ (in Θ_i and $\mathcal{L}_i$, respectively) then there are $\gamma_0, \gamma_1, \gamma_2 \in \Theta_{i+1}$ such that $\alpha \equiv_x \gamma_0 \equiv_y \gamma_1 \equiv_x \gamma_2 \equiv_y \beta$.*

3. *For every $\hat{\mathcal{L}} \subseteq_{lsl} \mathcal{L}_i$ there are* homogeneity interpolants *for Θ_i with respect to $\hat{\mathcal{L}}$ in Θ_{i+1}, i.e. for every $\alpha_0, \alpha_1, \beta_0, \beta_1 \in \Theta_i$ such that $\forall w \in \hat{\mathcal{L}}(\alpha_0 \equiv_w \alpha_1 \rightarrow \beta_0 \equiv_w \beta_1)$, there are $\gamma_0, \gamma_1 \in \Theta_{i+1}$ and $\hat{\mathcal{L}}$-homomorphisms $f, g, h : \Theta_i \to \Theta_{i+1}$ such that $f : \alpha_0, \alpha_1 \mapsto \beta_0, \gamma_1$, $g : \alpha_0, \alpha_1 \mapsto \gamma_0, \gamma_1$ and $h : \alpha_0, \alpha_1 \mapsto \gamma_0, \beta_1$.*

Proof. We first define the sequence $\mathcal{L}_i$ of slsls of $\mathcal{L}$ beginning with $\mathcal{L}_0$ which consists of the 0 and 1 of $\mathcal{L}$. We let the other elements of $\mathcal{L}$ be x_n for $n \geq 1$ and $\mathcal{L}_n$ be the (necessarily finite) slsl of $\mathcal{L}$ generated by $\{0, 1, x_1, \ldots, x_n\}$. As for Θ, we choose a countable set α_i and stipulate that $\Theta = \{\alpha_i | i \in \omega\}$. We begin defining the (values of) the α_i by setting $\alpha_0(x) = 0$ for all $x \in \mathcal{L}$ and $\alpha(0) = 0$ for all $\alpha \in \Theta$. We will now define Θ_n and the values of $\alpha \in \Theta_n$ (other than α_0) on the elements of $\mathcal{L}_n$ (other than 0) by recursion. For Θ_0 we choose a new element β of Θ and let $\Theta_0 = \{\alpha_0, \beta\}$ and set $\beta(1) = 1$. Given Θ_n and the values for its elements on $\mathcal{L}_n$ we wish to enlarge Θ_n to Θ_{n+1} and define the values of $\alpha(x)$ for $\alpha \in \Theta_{n+1}$ and $x \in \mathcal{L}_{n+1}$ so that the requirements of the Theorem are satisfied. To do this we prove a number of general extension theorems for usl representations in the Propositions below that show that we can make simple extensions to satisfy any particular meet or homogeneity requirement and also extend usl representations from smaller to larger slsls of $\mathcal{L}$. To be more specific, we first apply Proposition meetinterp 10.3.5 successively for each choice of $x \wedge y = z$ in $\mathcal{L}_n$ and $\alpha, \beta \in \Theta_n$ with $\alpha \equiv_z \beta$ choosing new elements of Θ to form Θ'_n extending Θ_n and defining them on $\mathcal{L}_n$ so that $\Theta'_n \restriction \mathcal{L}_n$ is an usl table for $\mathcal{L}_n$ containing Θ_n and the required meet interpolants for every such x, y, z, α and β. We then apply Proposition hominterp 10.3.6 successively for each $\hat{\mathcal{L}} \subseteq_{lsl} \mathcal{L}_n$ and each $\alpha_0, \alpha_1, \beta_0, \beta_1 \in \Theta_n$ such that $\forall w \in \hat{\mathcal{L}}(\alpha_0 \equiv_w \alpha_1 \rightarrow \beta_0 \equiv_w \beta_1)$ to get larger subset Θ''_n of Θ which we also define on $\mathcal{L}_n$ so as to have an usl table $\Theta''_n \restriction \mathcal{L}_n$ for $\mathcal{L}_n$ that has the required homogeneity interpolants and $\hat{\mathcal{L}}$-homomorphisms from Θ_n into Θ''_n for every such $\alpha_0, \alpha_1, \beta_0, \beta_1 \in \Theta_n$. Finally, we apply Proposition extend 10.3.4 to define the elements of Θ''_n on $\mathcal{L}_{n+1}$ and further enlarge it to our desired finite $\Theta_{n+1} \subseteq \Theta$ with all its new elements also defined on $\mathcal{L}_{n+1}$ so as to have an usl table of $\mathcal{L}_{n+1}$ with all the properties required by the Theorem. It is now immediate from the definitions that the union Θ of the Θ_n is an usl table of $\mathcal{L}$. ■

Notation 10.3.2 *If a finite $\hat{\mathcal{L}}$ is a slsl of $\mathcal{L}$, $\hat{\mathcal{L}} \subseteq_{lsl} \mathcal{L}$, and $x \in \mathcal{L}$ then we let $\hat{x}$ denote the least element of $\hat{\mathcal{L}}$ above x. The desired element of $\hat{\mathcal{L}}$ exists because $\hat{\mathcal{L}}$ is a slsl of $\mathcal{L}$ and so the infimum (in $\hat{\mathcal{L}}$ or, equivalently, in $\mathcal{L}$) of $\{u \in \hat{\mathcal{L}} | x \leq u\}$ is in $\hat{\mathcal{L}}$ and is the desired $\hat{x}$. As $\hat{\mathcal{L}}$ is finite it is also a lattice but join in $\hat{\mathcal{L}}$ may not agree with that in $\mathcal{L}$. We denote them by $\vee_{\hat{\mathcal{L}}}$ and $\vee_{\mathcal{L}}$ respectively when it is necessary to make this distinction.*

basichat **Lemma 10.3.3** *With the notation as above, $\hat{x} = x$ for $x \in \hat{\mathcal{L}}$ and so it is an idempotent operation. If $x \leq y$ are in $\mathcal{L}$ then $\hat{x} \leq \hat{y}$. If $x \vee_{\mathcal{L}} y = z$ are in $\mathcal{L}$ then $\hat{z} = \hat{x} \vee_{\hat{\mathcal{L}}} \hat{y}$.*

Proof. The first two assertions follow immediately from the definition of $\hat{x}$. The third is only slightly less immediate: $x, y \leq x \vee_{\mathcal{L}} y = z$ and so by the second assertion, $\hat{x}, \hat{y} \leq \hat{z}$ and so $\hat{x} \vee_{\hat{\mathcal{L}}} \hat{y} \leq \hat{z}$. For the other direction, note that as $x \leq \hat{x}$, $y \leq \hat{y}$, we have that $z = x \vee_{\mathcal{L}} y \leq \hat{x} \vee_{\mathcal{L}} \hat{y} \leq \hat{x} \vee_{\hat{\mathcal{L}}} \hat{y} \in \hat{\mathcal{L}}$ and so $\hat{z} \leq \hat{x} \vee_{\hat{\mathcal{L}}} \hat{y}$. ■

extend **Proposition 10.3.4** *If Θ is a finite usl table for $\hat{\mathcal{L}} \subseteq_{lsl} \mathcal{L}$ (finite) then there are extensions for each $\alpha \in \Theta$ to maps with domain $\mathcal{L}$ and finitely many further functions β with domain $\mathcal{L}$ such that adding them on to our extensions of the $\alpha \in \Theta$ provides an usl table Θ' of $\mathcal{L}$ with $\Theta \subseteq \Theta' \restriction \hat{\mathcal{L}}$. Moreover, these extensions can be found uniformly recursively in the given data (Θ, $\hat{\mathcal{L}}$ and $\mathcal{L}$).*

Proof. For $\alpha \in \Theta$ and $x \in \mathcal{L}$ set $\alpha(x) = \alpha(\hat{x})$. We first check that we have maintained the order and join properties required of an usl representation. If $x \leq y$ are in $\mathcal{L}$, $\alpha, \beta \in \Theta$ and $\alpha \equiv_y \beta$ then by definition $\alpha \equiv_{\hat{y}} \beta$ and so $\alpha \equiv_{\hat{x}} \beta$ as $\hat{x} \leq \hat{y}$ by Lemma basichat 10.3.3 and Θ's being an usl table of $\hat{\mathcal{L}}$. Thus, by definition, $\alpha \equiv_x \beta$ as required.

Next, if $x \vee_{\mathcal{L}} y = z$ are in $\mathcal{L}$ and $\alpha \equiv_{x,y} \beta$ we wish to show that $\alpha \equiv_z \beta$. Again by definition $\alpha \equiv_{\hat{x},\hat{y}} \beta$. By Lemma basichat 10.3.3, $\hat{x} \vee_{\hat{\mathcal{L}}} \hat{y} = \hat{z}$, so by Θ being an usl table for $\hat{\mathcal{L}}$, $\alpha \equiv_{\hat{z}} \beta$ and so by definition, $\alpha \equiv_z \beta$.

All that remains is to show that we can add on new maps with domain $\mathcal{L}$ that provide witnesses for the differentiation property for elements of $\mathcal{L} - \hat{\mathcal{L}}$ while preserving the order and join properties. This is a standard construction. For each pair $x \not\leq y$ (in $\mathcal{L}$ but not both in $\hat{\mathcal{L}}$) in turn we add on new elements $\alpha_{x,y}$ and $\beta_{x,y}$ with all new and distinct values at each $z \in \mathcal{L}$ except that they agree on all $z \leq x$ (and at 0, of course, have value 0). These new elements obviously provide the witnesses required for the differentiation property for an usl representation. It is easy to see that they also cause no damage to the order or join properties. There are no new nontrivial instances of congruences between them and the old ones in Θ (extended to $\mathcal{L}$). Among the new elements the only instances to consider are ones between $\alpha_{x,y}$ and $\beta_{x,y}$ for the same pair x, y and for lattice elements z less than or equal to x. As $\alpha_{x,y} \equiv_z \beta_{x,y}$ for all $z \leq x$, the order and join properties are immediate. ■

meetinterp **Proposition 10.3.5** *If $\alpha, \beta \in \Theta$, an usl table for a finite lattice $\mathcal{L}$, $\alpha \equiv_z \beta$ and $x \wedge y = z$ in $\mathcal{L}$ then there are $\gamma_0, \gamma_1, \gamma_2$ such that $\alpha \equiv_x \gamma_0 \equiv_y \gamma_1 \equiv_x \gamma_2 \equiv_y \beta$ and $\Theta \cup \{\gamma_0, \gamma_1, \gamma_2\}$ is still an usl table for $\mathcal{L}$. Moreover, these extensions can be found uniformly recursively in the given data.*

Proof. If $x \leq y$, there is nothing to be proved. Otherwise, the interpolants can be defined by letting $\gamma_0(w)$ be $\alpha(w)$ for $w \leq x$ and new values for $w \not\leq x$; $\gamma_1(w) = \gamma_0(w)$ for $w \leq y$ and new values otherwise; and $\gamma_2(w) = \beta(w)$ for $w \leq y$, $\gamma_2(w) = \gamma_1(w)$ if $w \leq x$ but $w \not\leq y$ and new otherwise. ■

hominterp **Proposition 10.3.6** *If $\hat{\mathcal{L}} \subseteq_{lsl} \mathcal{L}$, a finite lattice, and Θ is an usl table for $\mathcal{L}$ with $\alpha_0, \alpha_1, \beta_0, \beta_1 \in \Theta$ such that $\forall w \in \hat{\mathcal{L}}(\alpha_0 \equiv_w \alpha_1 \rightarrow \beta_0 \equiv_w \beta_1)$, then there is an usl table $\tilde{\Theta} \supseteq \Theta$ for $\mathcal{L}$ with $\gamma_0, \gamma_1 \in \tilde{\Theta}$ and $\hat{\mathcal{L}}$ homomorphisms $f, g, h : \Theta \rightarrow \tilde{\Theta}$ such that $f : \alpha_0, \alpha_1 \mapsto \beta_0, \gamma_1$, $g : \alpha_0, \alpha_1 \mapsto \gamma_0, \gamma_1$ and $h : \alpha_0, \alpha_1 \mapsto \gamma_0, \beta_1$. Moreover, these extensions can be found uniformly recursively in the given data.*

Proof. For each $\alpha \in \Theta$ and $x \in \mathcal{L}$ we set $f(\alpha)(x) = \beta_0(x)$ if $\alpha \equiv_{\hat{x}} \alpha_0$ and otherwise we let it be a new number that depends only on $\alpha(\hat{x})$, e.g. $\alpha(\hat{x})^*$. Note that which case of the definition applies for $f(\alpha)(x)$ depends only on $\alpha(\hat{x})$ and it can be an "old" value (i.e. one of some $\beta \in \Theta$) only in the first case. Thus, for $\alpha, \beta \in \Theta$,

$$\text{(a) } \alpha \equiv_{\hat{x}} \beta \Leftrightarrow f(\alpha) \equiv_x f(\beta) \text{ and (b) } f(\alpha) \equiv_x \beta \Rightarrow \alpha \equiv_{\hat{x}} \alpha_0 \Rightarrow f(\alpha) \equiv_x \beta_0. \qquad (10.1)$$ 1

Let $\Theta_1 = \Theta \cup f[\Theta]$. We claim that Θ_1 is an usl table for $\mathcal{L}$ and f is an $\hat{\mathcal{L}}$-homomorphism from Θ into Θ_1. That f is an $\hat{\mathcal{L}}$-homomorphism is immediate from the first clause in (10.1) and the fact (Lemma 10.3.3) that $\hat{x} = x$ for $x \in \hat{\mathcal{L}}$. We next check that Θ_1 satisfies the properties required of an usl representation. Of course, $f(\alpha)(0) = 0$ by definition for every α and differentiation is automatic as it extends Θ.

First, to check the order property for Θ_1 we consider any $x \le y$ in $\mathcal{L}$. As Θ is already an usl table for $\mathcal{L}$, it suffices to consider two cases for the pair of elements of Θ_1 which are given as congruent modulo y and show that in these two cases they are also congruent modulo x. The two cases are that (a) both are in $f[\Theta]$ and that (b) one is in $f[\Theta]$ and the other in Θ. Thus it suffices to consider any $\alpha, \beta \in \Theta$, assume that (a) $f(\alpha) \equiv_y f(\beta)$ or (b) $f(\alpha) \equiv_y \beta$ and prove that (a) $f(\alpha) \equiv_x f(\beta)$ and (b) $f(\alpha) \equiv_x \beta$, respectively. For (a), we have by (10.1) that $\alpha \equiv_{\hat{y}} \beta$ and so by the order property for Θ, $\alpha \equiv_{\hat{x}} \beta$. Thus $f(\alpha) \equiv_x f(\beta)$ by definition as required. As for (b), (10.1) tells us here that $\alpha \equiv_{\hat{y}} \alpha_0$ and $\beta \equiv_y f(\alpha) \equiv_y \beta_0$ (and therefore $\beta \equiv_x \beta_0$). Now by Lemma 10.3.3 $\alpha \equiv_{\hat{x}} \alpha_0$ so $f(\alpha) \equiv_x \beta_0$ and so $f(\alpha) \equiv_x \beta$ as required.

Next we verify the join property for $x \vee y = z$ in $\mathcal{L}$ and two elements of Θ_1 (not both in Θ) in the same two cases. For (a) we have that $f(\alpha) \equiv_{x,y} f(\beta)$ and so as above $\alpha \equiv_{\hat{x},\hat{y}} \beta$. Now by the join property in Θ and Lemma 10.3.3, $\alpha \equiv_{\hat{z}} \beta$ and so $f(\alpha) \equiv_z f(\beta)$ as required. For (b) using (10.1b) and Lemma 10.3.3 again we have that $f(\alpha) \equiv_{x,y} \beta \Rightarrow \alpha \equiv_{\hat{x},\hat{y}} \alpha_0 \Rightarrow \alpha \equiv_{\hat{z}} \alpha_0 \Rightarrow f(\alpha) \equiv_z \beta_0$ while it also tells us that $\beta \equiv_{x,y} f(\alpha) \equiv_{x,y} \beta_0$ as required. Note that clearly $f(\alpha_0) = \beta_0$. We let $\gamma_1 = f(\alpha_1)$ and so have the first function and (partial) extension of Θ required in the Proposition.

We now define h on Θ_1 as we did f on Θ using α_1 and β_1 in place of α_0 and β_0, respectively: $h(\alpha)(x) = \beta_1(x)$ if $\alpha \equiv_{\hat{x}} \alpha_1$ and otherwise we let it be a new number that depends only on $\alpha(\hat{x})$, e.g. $\alpha(\hat{x})^{**}$. Let $\Theta_2 = \Theta_1 \cup h[\Theta_1]$. As above, Θ_2 is an usl table for $\mathcal{L}$ and h is an $\hat{\mathcal{L}}$-homomorphism from Θ_1 (and so Θ) into Θ_2 taking α_1 to β_1. We let $\gamma_0 = h(\alpha_0)$ and so have the third function and (partial) extension of Θ required in the Proposition. As above in (10.1), we have for any $\alpha, \beta \in \Theta_1$ and $x \in \mathcal{L}$,

$$\text{(a) } \alpha \equiv_{\hat{x}} \beta \Leftrightarrow h(\alpha) \equiv_x h(\beta) \text{ and (b) } h(\alpha) \equiv_x \beta \Rightarrow \alpha \equiv_{\hat{x}} \alpha_1 \Rightarrow h(\alpha) \equiv_x \beta_1. \qquad (10.2) \quad \boxed{2}$$

Applying the second clause to $\gamma_0 = h(\alpha_0)$ and first to any $\beta \in \Theta_1$ and then, in particular to γ_1 we have

$$\text{(a) } \gamma_0 \equiv_x \beta \Rightarrow \alpha_0 \equiv_{\hat{x}} \alpha_1 \Rightarrow f(\alpha_1) = \gamma_1 \equiv_x \beta_0 \text{ and (b) } \gamma_0 \equiv_x \gamma_1 \Leftrightarrow \alpha_0 \equiv_{\hat{x}} \alpha_1. \qquad (10.3) \quad \boxed{3}$$

To see the right to left direction of the second clause, note that $\alpha_0 \equiv_{\hat{x}} \alpha_1$ implies that $\gamma_0 \equiv_x \beta_1$ and $\gamma_1 \equiv_x \beta_0$ by the definitions of h and f, respectively, while it also implies that $\beta_0 \equiv_{\hat{x}} \beta_1$ by the basic assumption of the Proposition. Thus, as Θ is an usl table of $\mathcal{L}$ and $x \le \hat{x}$, $\beta_0 \equiv_x \beta_1$ and $\gamma_0 \equiv_x \gamma_1$.

Finally, we define g on $\alpha \in \Theta_2$ by setting $g(\alpha)(x) = \gamma_0(x)$ if $\alpha \equiv_{\hat{x}} \alpha_0$. If $\alpha \not\equiv_{\hat{x}} \alpha_0$ but $\alpha \equiv_{\hat{x}} \alpha_1$ then $g(\alpha)(x) = \gamma_1(x)$. Otherwise, we let $g(\alpha)(x)$ be a new number that depends only on $\alpha(\hat{x})$, e.g. $\alpha(\hat{x})^{***}$. Note that if $\alpha \equiv_{\hat{x}} \alpha_1$ then we always have $g(\alpha) \equiv_x \gamma_1$ as if

$\alpha \equiv_{\hat{x}} \alpha_0$ as well then, by (10.3b), $\gamma_0 \equiv_x \gamma_1$. Thus $g(\alpha_0) = \gamma_0$ and $g(\alpha_1) = \gamma_1$ as required. It is also obvious that g is an $\hat{\mathcal{L}}$-homomorphism of Θ_2 (and so Θ) into $\Theta_3 = \Theta_2 \cup g[\Theta_2]$ as by definition and Lemma 10.3.3, $\alpha \equiv_{\hat{x}} \beta \Rightarrow g(\alpha) \equiv_{\hat{x}} g(\beta)$ for any $x \in \mathcal{L}$. Indeed, for any $\alpha, \beta \in \Theta_2$ and $x \in \mathcal{L}$

$$\alpha \equiv_{\hat{x}} \beta \Leftrightarrow g(\alpha) \equiv_x g(\beta). \tag{10.4}$$ $\boxed{4}$

To see the right to left direction here, note that if either of $g(\alpha)$ or $g(\beta)$ is new for g at x (i.e. of the form $\delta(\hat{y})^{***}$) then clearly both are. In this case, $\alpha \equiv_{\hat{x}} \beta$ by definition. Otherwise, either they are both congruent to α_0 or both to α_1 and so congruent to each other mod $\hat{x}$. The point here is that if one is congruent to α_0 and the other to α_1 but not α_0 at $\hat{x}$ then by definition $\gamma_0 \equiv_x \gamma_1$ and so by (10.3b), $\alpha_0 \equiv_{\hat{x}} \alpha_1$ for a contradiction.

Thus we only need to verify that Θ_3 is an usl table of $\mathcal{L}$. We consider any $\alpha, \beta \in \Theta_2$ and divide the verifications into cases (a) and (b) as before with the former considering $g(\alpha)$ and $g(\beta)$ and the latter $g(\alpha)$ and β. These cases may then be further subdivided.

We begin with the order property and so $x \leq y$ in $\mathcal{L}$.

(a) If $g(\alpha) \equiv_y g(\beta)$ then, by (10.4), $\alpha \equiv_{\hat{y}} \beta$ and so $\alpha \equiv_{\hat{x}} \beta$ as $\hat{x} \leq \hat{y}$ (Lemma 10.3.3) and Θ_2 is an usl table of $\mathcal{L}$. Thus, again by (10.4) $g(\alpha) \equiv_x g(\beta)$ as required.

(b) If $g(\alpha) \equiv_y \beta$ then by definition they are congruent modulo y to γ_i (for some $i \in \{0,1\}$) and α is congruent to α_i at $\hat{y}$. Thus $\alpha \equiv_{\hat{x}} \alpha_i$ as $\hat{x} \leq \hat{y}$ and Θ_2 is an usl table so $g(\alpha) \equiv_x \gamma_i$ by definition. Similarly, as $x \leq y$, $\beta \equiv_x \gamma_i$ as well.

Now for the join property for $x \vee y = z$ in $\mathcal{L}$.

(a) If $g(\alpha) \equiv_{x,y} g(\beta)$ then, as above, $\alpha \equiv_{\hat{x},\hat{y}} \beta$. As $\hat{x} \vee \hat{y} = \hat{z}$ by Lemma 10.3.3 and Θ_2 is an usl representation, $\alpha \equiv_{\hat{z}} \beta$ and so by (10.4) $g(\alpha) \equiv_z g(\beta)$ as required.

(b) If $g(\alpha) \equiv_{x,y} \beta$ then again $\alpha \equiv_{\hat{x}} \alpha_i$ and $\alpha \equiv_{\hat{y}} \alpha_j$ for some $i, j \in \{0,1\}$ and $g(\alpha) \equiv_x \beta \equiv_x \gamma_i$ while $g(\alpha) \equiv_y \beta \equiv_y \gamma_j$. If $i = j$ then $\alpha \equiv_{\hat{x},\hat{y}} \alpha_i$ and so $\alpha \equiv_{\hat{z}} \alpha_i$ and $g(\alpha) \equiv_z \gamma_i \equiv_z \beta$ as required.

On the other hand, suppose, without loss of generality, that $(*)$ $\alpha \equiv_{\hat{x}} \alpha_0$ and so $\beta \equiv_x g(\alpha) \equiv_{\hat{x},x} \gamma_0 = h(\alpha_0)$ while $\alpha_0 \not\equiv_{\hat{y}} \alpha \equiv_{\hat{y}} \alpha_1$ and so $\beta \equiv_y g(\alpha) \equiv_{\hat{y},y} \gamma_1 = f(\alpha_1)$. If $\beta \in \Theta_1$ then by (10.4a) $\alpha_0 \equiv_{\hat{x}} \alpha_1$ and so $\alpha \equiv_{\hat{x}} \alpha_1$. As our assumption is that $\alpha \equiv_{\hat{y}} \alpha_1$ we have (by the join property in Θ_2) that $\alpha \equiv_{\hat{z}} \alpha_1$ and so $g(\alpha) \equiv_z \gamma_1$. As $\alpha_0 \equiv_{\hat{x}} \alpha_1$ (10.3b) tells us that $\gamma_0 \equiv_x \gamma_1$. Our assumptions then say that $\beta \equiv_{x,y} \gamma_1$ and so $\beta \equiv_z \gamma_1$ as required. Thus we may assume that $\beta = h(\delta)$ for some $\delta \in \Theta_1$.

We now have $h(\delta) = \beta \equiv_x g(\alpha) \equiv_x \gamma_0 = h(\alpha_0) \in \Theta_1$ and so by (10.2a) applied to $h(\delta) \equiv_x h(\alpha_0)$ with δ for α and α_0 for β we see that $\delta \equiv_{\hat{x}} \alpha_0$. We also have $h(\delta) = \beta \equiv_y g(\alpha) \equiv_{\hat{y},y} \gamma_1 = f(\alpha_1)$. Applying (10.2b) to $h(\delta) \equiv_y \gamma_1$ with δ for α and $\gamma_1 \in \Theta_1$ for β, we see that $\delta \equiv_{\hat{y}} \alpha_1$and $h(\delta) \equiv_y \beta_1$ and so $\beta_1 \equiv_y \gamma_1 = f(\alpha_1)$. Now applying (10.1b) with α_1 for α and $\beta_1 \in \Theta$ for β, we have that $\alpha_1 \equiv_{\hat{y}} \alpha_0$. As this contradicts $(*)$, we are done. ■

2qtth 10.4 Decidability of two quantifier theory

3qtth 10.5 Undecidability of three quantifier theory.

Also two quantifier with $\vee$ and $\wedge$.?

Other results establishing borderlines in other languages e.g. with jump? If so in different chapter/section?

comments on what known below $\mathbf{0}'$

Chapter 11

Π^0_1 Classes

11.1 Binary trees

We now return to the basic our basic notion of a tree as a downward closed subset of $\omega^{<\omega}$. In this context we use T_σ to denote the subtree of T consisting of all strings ρ compatible with σ: $T_\sigma = \{\rho | \rho \subseteq \sigma \text{ or } \sigma \subseteq \rho\}$. Recall that the sets of paths in such trees are the closed sets in Baire space ω^ω. In this chapter we will be primarily concerned with infinite binary trees, i.e. the infinite downward closed subsets T of $2^{<\omega}$. We endow each binary tree with a left to right partial order as well as the order of extension. It is specified by the lexicographic order on strings so σ is to the left of τ, $\sigma <_L \tau$ if $\sigma(n) < \tau(n)$ for the least n such that $\sigma(n) \neq \tau(n)$ if there is one. (This order extends in the obvious way to one of 2^ω which we also call the left to right or lexicographic order.) The sets of paths $[T] = \{A \in 2^\omega : \forall n(A \upharpoonright n \in T)\}$ through these trees are precisely the nonempty closed subsets of Cantor space, 2^ω.

Exercise 11.1.1 *For any binary tree T, $[T]$ is a closed set in Cantor space.*

??Prove??

To see that every closed subset of 2^ω is of the form $[T]$ for some tree T, consider the open sets in 2^ω. They are all unions of basic (cl)open sets of the form $[\sigma] = \{f \in 2^\omega | \sigma \subseteq f\}$ for $\sigma \in 2^{<\omega}$. So given any closed set $\mathcal{C}$ its complement $\bar{\mathcal{C}}$ is a union of such neighborhoods. Let $T = \{\sigma | [\sigma] \subsetneq \bar{\mathcal{C}}\} = \{\sigma | [\sigma] \cap \mathcal{C} \neq \emptyset\}$. It is clear that T is downward closed. If $f \in \mathcal{C}$ and $\sigma \subseteq f$ then clearly $\sigma \in T$ and so $f \in [T]$. On the other hand if $f \in [T]$ and $\sigma \subseteq f$ then $\sigma \in T$ and so the closed set $[\sigma] \cap \mathcal{C} \neq \emptyset$. As Cantor space is compact $\cap\{[\sigma] \cap \mathcal{C} | \sigma \subseteq f\}$ is nonempty and only f can be in it so $f \in \mathcal{C}$ as required. Note that, by König's lemma (Lemma $\overset{\text{KL}}{4.2.4}$), $\mathcal{C}$ is nonempty if and only if T is infinite.

??Move this material to Trees section and recall here??

In this chapter we want to investigate the recursive versions of these two notions.

Definition 11.1.2 *A class $\mathcal{C} \subseteq 2^\omega$ is* effectively closed *if it is of the form $[T]$ for a recursive binary tree T.*

We can also characterize the effectively closed sets in terms of the complexity of their definition. We use the same notation based on the arithmetic hierarchy for classes of sets or functions as we did for individual sets and functions.??say more now or before Go back and check definitions for Σ^A_n especially Σ_0 and how interpret for σ in place of A ...bounded quantifiers??

Definition 11.1.3 *A class $\mathcal{C} \subseteq 2^\omega$ of sets is Σ^0_n (Π^0_n) if there is a Σ^0_n (Π^0_n) formula $\varphi(X)$ with one free set variable X such that $\mathcal{C} = \{A | \mathbb{N} \vDash \varphi(A)\}$. Similarly for classes $\mathcal{F} \subseteq \omega^\omega$ of functions and formulas with one free function variable.*

The primary connection with trees is the following Proposition.

pi01trees **Proposition 11.1.4** *The Π^0_1 classes of sets are precisely the sets of paths through recursive binary trees. Again, the nonempty classes correspond to the infinite recursive binary trees. Moreover, there is a recursive procedure that takes an index for a Π^0_1 formula to one for a recursive tree T such that $[T]$ is the corresponding Π^0_1 class.*

Proof. If T is a recursive binary tree then $[T] = \{A \in 2^\omega : \forall n(A \upharpoonright n \in T)\}$ is clearly a Π^0_1 class. If T is infinite, $[T]$ is nonempty by König's lemma while if T is finite $[T]$ is clearly empty. For the other direction consider any Π^0_1 class $\mathcal{P} = \{A : \forall x R(A, x)\}$ for a Σ^A_0 relation R. Let $T = \{\sigma \in 2^{<\omega} | \neg(\exists x < |\sigma|)\neg R(\sigma, x)\}$ where we understand that we are thinking of σ as representing an initial segment of A. Formally we replace $t \in A$ by $\sigma(t) = 1$ and declare the formula $R(\sigma, x)$ false if some term $t > |\sigma|$ occurs in it as described in ??. It is then immediate that $P = [T]$ and that an index for T as recursive function is given uniformly in the index for R as a Σ^A_0 formula. If P is nonempty, T has an infinite path and so is itself infinite. Otherwise, T is finite. ■

Exercise 11.1.5 *The Π^0_1 classes of functions are precisely the sets of paths through recursive trees (on $\omega^{<\omega}$).*

We can now index the Π^0_1 classes (of sets) by either the indices of the Π^0_1 formulas or of the trees derived from them as in the proof of Proposition pi01trees 11.1.4 as partial recursive functions which are actually total. A natural question then is how hard is to tell if a recursive tree is infinite or a Π^0_1 class is nonempty. It might seem at first that these properties are Π^0_2 and so only recursive in $0''$. If we know that the tree is recursive as we do for the trees derived uniformly from Π^0_1 classes, however, then the question is actually uniformly (on indices) recursive in $0'$. This observation depends on the compactness of Cantor space and plays a crucial role in almost every argument in the rest of this chapter.

fin0' **Lemma 11.1.6** *If T is a recursive binary tree (say with index i so $T = \Phi_i$) then T being finite is a Σ^0_1 property (of i). Thus we can decide if T is finite or infinite recursively in $0'$. Indeed, T is finite if and only if there is an n such that $\sigma \notin T$ for every σ of length n.*

Proof. Clearly, T is finite if and only if there is an n such that $\sigma \notin T$ for every σ of length n. Clearly this is a Σ^0_1 property for any recursive binary tree and the associated Σ^0_1 formula is given uniformly in a recursive index for T. ■

While deciding if a given recursive binary tree is infinite or a Π^0_1 class nonempty requires $0'$, we can actually make a recursive list of the nonempty Π^0_1 classes and so one of corresponding infinite recursive binary trees (up to the set of paths on T).

recindpi01 **Exercise 11.1.7** *There is a uniformly recursive list of the nonempty Π^0_1 classes in the sense that there is a recursive set Q such that, for each $e \in Q$, Φ_e is (the characteristic function of) an infinite binary tree T_e and for every nonempty Π^0_1 class $\mathcal{C}$ there is an e such that $\mathcal{C} = [\Phi_e] = [T_e]$. Hint: For each e consider the r.e. set W_e viewing its elements as binary strings σ. We now form a recursive tree T_e by putting in the empty string at stage 0 and then at stage $s > 0$ exactly those strings τ of length s with no $\sigma \subseteq \tau$ in $W_{e,s}$ unless there are none (equivalently $\cup\{[\sigma]|\sigma \in W_{e,s}\} = 2^\omega$), in which case we declare all immediate successors of strings in T_e of length $s-1$ to be in T_e as well. Note that T_e is uniformly recursive. For one direction prove that each T_e is infinite (and so $[T_e]$ is a nonempty Π^0_1 class). For the other direction, if $\mathcal{C}$ is a nonempty Π^0_1 class then the set $\{\sigma|[\sigma] \cap \mathcal{C} = \emptyset\}$ is r.e. and so equal to some W_e. Now show that $[T_e] = \mathcal{C}$.*

We now present some important Π^0_1 classes.

Example 11.1.8 *$DNR_2 = \{f \in 2^\omega : f$ is $DNR\}$. Recall that DNR means $f(e) \neq \Phi_e(e)$. In other words, $\forall e \forall s \neg(f(e) = \Phi_{e,s}(e))$. Thus, DNR_2 is a Π^0_1 class.*

Example 11.1.9 *Let H be any recursively axiomatizable consistent theory. The class $\mathcal{C}_H$=$\{f \in 2^\omega : f$ is a complete extension of $H\}$ is a Π^0_1 class. By the assertion that f "is a complete extension of H" we mean that we have a recursive coding (Gödel numbering) φ_n of the sentences of H such that $T_f = \{\varphi_n|f(n) = 1\}$ is deductively closed, contains all the axioms of H and is consistent in the sense that there is no φ such that f assigns 1 (true) to both φ and $\neg\varphi$. The only point to make about this being a Π^0_1 class is perhaps the requirement that T_f be deductively closed. This says that for all finite sets Φ of sentences and each sentence φ_k and proof p, if p is a proof that $\Phi \vdash \varphi$ and $f(n) = 1$ for every $\varphi_n \in \Phi$ then $f(k) = 1$.*

Example 11.1.10 *If A, B are disjoint r.e. sets, then the class $S(A,B) = \{C : C \supset A \ \&\ C \cap B = \emptyset\}$ of* separating sets *C (for the pair (A,B)) is a Π^0_1 class as is obvious from its definition: $S = \{C : \forall n(n \in A \rightarrow n \in C \ \&\ n \in B \rightarrow n \notin C)\}$. Since $A, B \in \Sigma^0_1$ this is a Π^0_1 formula.*

We can view a Π^0_1 class as the solution set to the problem of finding an f that satisfies the defining condition for the class. Equivalently, the problem is finding a path f through the corresponding tree T. For the above examples the problems are to construct a DNR_2 function, a complete consistent extension of H and a separating set for A

and B, respectively. If we choose our theory H and our disjoint r.e. sets A and B correctly then the three problems and so the Π^0_1 classes (and the $[T]$ for the corresponding trees) are equivalent in the sense that a solution to (path through) any one of them computes a solution for (path in) each of the others. Suitable choices for H and (A, B) are Peano arithmetic, PA, ??define before?? and (V_0, V_1) where $V_0 = \{e : \Phi_e(e) = 0\}$ and $V_1 = \{e : \Phi_e(e) = 1\}$. For these choices, the problems are also universal in the sense that a solution to any one of them computes a path through any infinite recursive binary tree and hence a solution to any problem specified by a nonempty Π^0_1 class.

Theorem 11.1.11 *If T is an infinite recursive binary tree and f is a member of any of the three Π^0_1 classes DNR_2, $\mathcal{C}_{PA}$ or $S(V_0, V_1)$ described above then there is a path $g \in [T]$ with $g \leq_T f$.*

Proof. We first prove the theorem for $S(V_0, V_1)$. Suppose T is an infinite recursive binary tree. We begin by defining disjoint r.e. sets A and B such that any $f \in S(A, B)$ computes a path in T. We then show how to compute a path in (any) $S(A, B)$ from one in $S(V_0, V_1)$.

We know that $\{\sigma | T_\sigma \text{ is finite}\}$ is r.e. so suppose it is W_e. We let $A_0 = \{\sigma | \exists s(\sigma\hat{}0 \in W_{e,s}$ & $\sigma\hat{}1 \in W_{e,s}\}$ (the σ such that we "see" that $T_{\sigma\hat{}0}$ is finite before we "see" that $T_{\sigma\hat{}1}$ is finite) and $A_1 = \{\sigma | \exists s(\sigma\hat{}1 \in W_{e,s}$ & $\sigma\hat{}0 \in W_{e,s}\}$ (the σ such that we "see" that $T_{\sigma\hat{}1}$ is finite before we "see" that $T_{\sigma\hat{}0}$ is finite). It is clear that $A_0 \cap A_1 = \emptyset$. Let $C \in S(A_0, A_1)$ and define D a path in T by recursion. We begin with $\emptyset \in D$. If $\sigma \in D$ then we put $\sigma\hat{}C(\sigma)$ into D. We now argue by induction that if $\sigma \in D$ then T_σ is infinite: If T_σ is infinite then at least one of $T_{\sigma\hat{}0}$ and $T_{\sigma\hat{}1}$ is infinite. If both are infinite there is nothing to prove so suppose that $T_{\sigma\hat{}0}$ is finite but $T_{\sigma\hat{}1}$ is infinite. In this case, it is clear from the definition that $\sigma \in A_0$ and so $C(\sigma) = 1$ and we put $\sigma\hat{}1$ into D to verify the induction hypothesis. In the other case, $\sigma \in A_1$, $C(\sigma) = 0$ and we put $\sigma\hat{}0$ into D with $T_{\sigma\hat{}0}$ infinite as required.

Now we see how to compute a $C \in S(A_0, A_1)$ from any $D \in S(V_0, V_1)$. By the $s{-}m{-}n$ theorem ?? there is a recursive functions h such that $\forall n(n \in A_i \Leftrightarrow h(n) \in V_i)$. We now let $C(n) = D(h(n)$. It is easy to see that $C \in S(A_0, A_1)$ as required. Thus $S(V_0, V_1)$ is universal in the desired sense.

We now only have to prove that we can compute a member of $S(V_0, V_1)$ from any DNR_2 function f and from any complete extension P of PA. For the first, simply note that if $f \in DNR_2$ then $f \in S(V_0, V_1)$: If $e \in V_0$ then $\Phi_e(e) = 0$ and so $f(e) = 1$ as required. On the other hand, $e \in V_1$ then $\Phi_e(e) = 1$ and so $f(e) = 0$ as required.

Finally, suppose P is complete extension of PA. Define $C(n) = 1$ if P declares the sentence $\exists s(n \in V_{0,s}$ & $\forall t < s(n \notin V_{1,s}))$ to be true and 0 otherwise. Note that if $n \in V_0$ then there is a least $s \in \mathbb{N}$ such that $n \in V_{0,s}$. This fact is then provable in PA (computation is essentially a proof). Similarly, for each $t < s$, $n \notin V_{1,t}$ since $n \in V_0$ and so $C(n) = 1$ as required. On the other hand, if $n \in V_1$ then there is a least $s \in \mathbb{N}$ such that $n \in V_{1,s}$ and for each $t < s$, $n \notin V_{0,t}$ since $n \in V_1$ and so PA proves that

$\exists s(n \in V_{1,s} \ \&\ \forall t < s(n \notin V_{0,s}))$. As P is a consistent extension of PA, it cannot then prove that $\exists s(n \in V_{0,s} \ \&\ \forall t < s(n \notin V_{1,s}))$ and so $C(n) = 0$ as required.

Exercise 11.1.12 *Show that the degree classes* $\mathbf{DNR}_2$*,* $\mathbf{C}_{PA}$ *and* $\mathbf{S}(\mathbf{V}_0, \mathbf{V}_1)$ *consisting of the degrees in each of the corresponding* Π^0_1 *classes are all the same.*

■

As every DNR function is obviously nonrecursive (Proposition dnrnotrec 3.0.5), none of these three classes have recursive members. So in particular there are no recursive complete extension of PA and there is no recursive separating set for (V_0, V_1).

Thinking of Π^0_1 classes as problems that ask for solutions, the natural question is how complicated must solutions be or how simple can they be. In the (in some sense uninteresting) case that there is only one path in T (or only finitely many) we can say everything about their degrees.

Proposition 11.1.13 *If a recursive binary tree T has single path that path is recursive. In fact, any isolated path ??define?? on a recursive tree is recursive.*

In general for arbitrary T one easy answer to the question is that there are always solutions recursive in $0'$.

rec0' **Exercise 11.1.14** *Show that every nonempty* Π^0_1 *class has a member recursive in* $0'$*. Hint: it is immediate for the separating classes.*

It is not hard to say a bit more.

redegree **Proposition 11.1.15** *Every infinite recursive binary tree T has a path of r.e. degree. In fact, the leftmost path P in T has r.e. degree.*

Proof. ■

We, in fact, can significantly improve the result of Exercise rec0' 11.1.14. The Low Basis Theorem below (Theorem lowb 11.1.18) gives the best answer with the notion of simplicity of the desired solution measured by its jump class. It is called a *basis theorem* as we say that a class $\mathcal{C}$ is a *basis* for a collection of problems (sets) if every problem (set) in the collection has a solution (member) in $\mathcal{C}$. Theorem 0'domb 11.1.19 gives another basis result in terms of domination properties and Theorem pi01coneav 11.1.21 one in terms of solutions not computing given (nonrecursive) sets.

To prove each of these theorems we use the notion of forcing $\mathcal{P}$ whose conditions are basically infinite recursive binary trees T with usual notion of subtree as extension (simply a subset). To make the definition of our required function V recursive, we explicitly specify a stem τ for each tree such that every $\rho \in T$ is compatible with σ. Thus our conditions p are pairs (T, τ) with T an infinite recursive binary tree and $\tau \in T$ such that $(\forall \rho \in T)(\rho \subseteq \tau$ or $\tau \subseteq \rho)$. We say that $(T, \tau) \leq_{\mathcal{P}} (S, \sigma)$ if $T \subseteq S$ and $\tau \supseteq \sigma$.

Of course, $V((T,\tau)) = \tau$. If $p = (T,\tau)$ and $\sigma \supseteq \tau$, we use p_σ to denote the condition (T_σ, σ).

The complexity of this notion of forcing depends on the representation or indexing used for the infinite recursive binary trees. While,at one end we could use the recursive listing of Exercise recindpi01 11.1.7, it would then be more difficult to describe various operations on trees that determine subtrees in the natural sense but do not obviously produce an index of the type required. In this case we would also want to define the subtree relation $T \subseteq S$ in terms of $[T] \subseteq [S]$ which would then be a Π^0_2 relation (Exercise subtreepaths 11.1.16) and so only recursive in $0''$.

ubtreepaths **Exercise 11.1.16** *If e and i are indices for infinite binary recursive trees T and S then the relation $[T] \subseteq [S]$ is Π^0_2, and, in fact,it is Π^0_2 complete.*

At the other end, we can simply use indices for recursive functions that define infinite binary trees. While this set is only recursive in $0''$ (because it takes $0''$ to decide if an index is one for a recursive tree), operations on trees become easy to implement on the indices. On this set of indices, the standard subtree relation $T \subseteq S$ is then Π^0_1 and so recursive in $0'$. We adopt this representation of trees for our notion of forcing. In fact, while the notion of forcing is then only $0''$-recursive, some of what we want to do can be done recursively in $0'$ by analyzing the required density functions. As an example, we have the following Lemma.

Lemma 11.1.17 *There is a density function f for the class $V_n = \{(T,\tau)|\ |\tau| \geq n\}$ of dense sets in $\mathcal{P}$ which is recursive in $0'$.*

Proof. Given $p = (T,\tau) \in P$ and $n \in \mathbb{N}$, Lemma fin0' 11.1.6 tells us that we can find a $\sigma \in T$ $(\sigma \supseteq \tau)$ of length $m \geq n$ such that T_σ is infinite. Clearly $p_\sigma = (T_\sigma, \sigma) \in P$ and $V(p_\sigma) \geq n$. ■

lowb **Theorem 11.1.18 (Low Basis Theorem, Jockusch and Soare)** *If T is a recursive infinite binary tree then it has a low path, i.e. there is a $G \in [T]$ with $G' \equiv_T 0'$. Equivalently, if $\mathcal{C}$ is a nonempty Π^0_1 class, then it has a low member. Moreover, we can compute such a path uniformly recursively in $0'$ and the index for T or the class.*

Proof. As usual we want to show that the sets of conditions deciding the jump ($D_n = \{p|\Phi_n^{V(p)}(n)\downarrow$ or $(\forall q \leq_{\mathcal{P}} p)(\Phi_n^{V(q)}(n)\uparrow)\}$) are dense and provide a density function $f \leq_T 0'$ that also tells us in which way $f(p,n)$ is in D_n. By Lemma meetdense 6.1.18 starting with condition $p_0 = (T, \emptyset)$ we can meet these sets as well as the V_n by a generic sequence recursive in $0'$ and so construct a $G \in [T]$ with $G' \equiv_T 0'$ as required.

Given an $p = (T,\tau) \in P$ and an n, we cannot use our usual strategy of asking for a $\sigma \in T$ $(\sigma \supseteq \tau)$ such that $\Phi_n^\sigma(n)\downarrow$ and then taking say p_σ as $f(p,n)$ because T_σ may be finite. Instead we ask if $\hat{T} = \{\sigma \in T|\Phi_n^\sigma(n)\uparrow\}$ is infinite. This question can be answered by $0'$ by Lemma fin0' 11.1.6. If so, we let $f(p,n) = (\hat{T},\tau)$ and note that we have satisfied the second clause of the definition of D_n as well as guaranteed that $\Phi_n^G(n)\uparrow$ for every $G \in [\hat{T}]$ including, of course, the generic G we are constructing. If not, then clearly there is a $k \geq |\tau|$ such that $\Phi_n^\sigma(n)\downarrow$ for every $\sigma \in T$ of length k. T_σ must be infinite for one of these σ as T is infinite. Again by Lemma fin0' 11.1.6, $0'$ can find such a σ and we then set $f(p,n) = p_\sigma$. In this case, it is clear that we have satisfied the first clause of D_n and $\Phi_n^G(n)\downarrow$ for every $G \in [T_\sigma]$.

The assertion about members of the corresponding Π_1^0 classes as well as the uniformity claim in the theorem are now immediate. ■

Note that we cannot make a similar improvement to Proposition redegree 11.1.15. Any element of DNR_2, C_{PA} or $S(V_0,V_1)$ of r.e. degree has degree $0'$.??

We next give a different answer to how simple a path we can construct on an arbitrary infinite recursive binary tree. Now the notion of simplicity is specified in terms of domination properties.

0'domb **Theorem 11.1.19 (0′-dominated Basis Theorem)** *If T is an infinite recursive binary tree, then there is an $G \in [T]$ such that every $f \leq_T G$ is dominated by some recursive function.*

Proof. We use the same notion of forcing with new dense sets. In place of the D_n we have $E_n = \{(T,\tau) | (\exists x)(\forall \sigma \in T)(\Phi_n^\sigma(x)\uparrow \text{ or } (\forall x)(\exists k)(\forall \sigma \in T)_{|\sigma|=k}(\Phi_n^\sigma(x)\downarrow)\}$. To see that the E_n are dense consider any condition $p = (T,\tau)$. If there is an x such that $S = \{\sigma \in T | \Phi_n^\sigma(x)\uparrow\}$ is infinite then choose such an x and S. The desired extension of p in E_n is then (S,τ). Note that in this case, $\Phi_n^G(x)\uparrow$ for any $G \in [S]$. If there is no such x, then, by Lemma fin0' 11.1.6, $p = (T,\tau)$ satisfies the second clause in E_n and is itself the witness to density. Note that in this case Φ_n^G is total for any generic G. Indeed, we can now also define a recursive function h which dominates Φ_e^G for any $G \in [T]$: To compute $h(x)$ find a k such that $(\forall \sigma \in T)_{|\sigma|=k}(\Phi_n^\sigma(x)\downarrow$. This is a recursive procedure since by our case assumption there is always such a k. Now set $h(x) = \max\{\Phi_n^\sigma(x) | \sigma \in T \text{ and } |\sigma| = k\} + 1$. This function clearly dominates Φ_n^G for any $G \in [T]$. ■

Exercise 11.1.20 *Show that we may find a G as in Theorem 0'domb 11.1.19 with $G'' \leq_T 0''$.*

We next turn to finding paths in trees which are simple in the sense that they do not compute some given (nonrecursive) set C or, more generally, any of some countable collection C_i of nonrecursive sets.

pi01coneav **Theorem 11.1.21 (Cone Avoidance, Jockusch and Soare)** *If T is an infinite recursive binary tree and $\{C_i\}$ is a sequence of nonrecursive sets, there is an $A \in [T]$ such that $C_i \not\leq_T A$ for all i.*

Proof. We modify the proof of density of the E_n of Theorem 0'domb 11.1.19 to get $E_{n,m}$ that guarantee that $\Phi_n^G \neq C_m$. We let $E_{n,m} = \{(T,\tau) | (\exists x)(\forall \sigma \in T)(\Phi_n^\sigma(x)\uparrow \text{ or } (\exists x)(\Phi_n^\tau(x)\downarrow \neq C_m(x)) \text{ or } (\forall x)(\exists k)(\forall \sigma_0, \sigma_1 \in T)_{|\sigma_0|=k=|\sigma_1|}(\Phi_n^{\sigma_0}(x)\downarrow = \Phi_n^{\sigma_1}(x)\downarrow)\}$. Given any condition

(T, τ) we first extend it to $q = (S, \sigma) \in E_n$. If we satisfy the first clause of E_n we satisfy the same clause in $E_{n,m}$. Otherwise, we satisfy the second clause of E_n. We now ask if there are ρ_0,$\rho_1 \in S$ with $\rho_i \supseteq \sigma$ and an x such that the S_{ρ_i} are infinite and $\Phi_n^{\rho_0}(x) \downarrow \neq \Phi_n^{\rho_1}(x) \downarrow$. If so, we choose $i \in \{0, 1\}$ such that $\Phi_n^{\rho_i}(x) \neq C_m(x)$ and take q_{ρ_i} as our extension of q (and so of p) which gets into $E_{n,m}$ by satisfying the second clause. If not, we claim that q itself satisfies the third clause of $E_{n,m}$ and that there is a recursive function h such that $\Phi_n^G = h$ for every $G \in [S]$. As for q satisfying the third clause of $E_{n,m}$, consider any x and note that it already satisfies the second clause of E_n. If there were infinitely many k such there are $\sigma_0, \sigma_1 \in T$ of length k with $\Phi_n^{\sigma_0}(x) \downarrow \neq \Phi_n^{\sigma_1}(x) \downarrow$ then we would have been in the previous case as there would then be infinitely many $\sigma \in T$ with $\Phi_n^{\sigma}(x) \downarrow \neq C_m(x)$. Thus we may define $h(x)$ by finding a k as in the third clause of $E_{n,m}$ and setting $h(x) = \Phi_n^{\sigma}(x)$ for any σ in S of length k. We then have that $\Phi_n^G = h$ for every $G \in [S]$. As C_m is not recursive,$\Phi_n^G \neq C_m$ for any $G \in [S]$ and so we also satisfy the requirements of the theorem. ■

Exercise 11.1.22 *Show that we may construct a G as required in Theorem* pi01coneav *11.1.21 such that $G \leq_T 0'' \oplus (\oplus_i C_i)$ and indeed uniformly.*

Exercise 11.1.23 *For one nonrecursive C instead of a countable set of C_i show that we may construct a G as required in Theorem* pi01coneav *11.1.21 such that $G \leq_T 0''$ (but without the uniformity). Hint: use the following exercise.*

Exercise 11.1.24 *Prove that for any infinite recursive binary tree T there are $G_0, G_1 \in [T]$ such that any $C \leq_T G_0, G_1$ is recursive. Moreover, we may find such G_i with $G_i'' \equiv_T 0''$.*

Exercise 11.1.25 *Nonempty Π^0_1 classes such as DNR_2 that have no recursive member are called* special *Π^0_1 classes. Prove that any such class has $2^{\aleph_0}$ many members.*

Exercise 11.1.26 *Strengthen some of previous theorems producing a path in T with some property to producing $2^{\aleph_0}$ many if T is special.*

11.2 Finitely branching trees

Also trees recursive in A (f). Relativizations.

Finitely branching trees, f-bounded, (recursively bounded) essentially the same as binary (recursive) binary trees relativize results to f.

Given a recursive recursively bounded tree can get recursive binary tree which has same paths up to degree by padding.

The sets of paths through infinitely branching trees $T \subseteq \omega^{<\omega}$ correspond to closed sets in Baire space. Even for recursive trees finding paths is much more complicated in this setting. Whether such trees even have paths is a Π^1_1 complete question. As for a basis theorem, one says that if there is a path then there is one recursive in the complete Π^1_1 set $\mathcal{O}$.??

Reference for low basis theorem: Jockusch, Soare " Degrees of Members of Π^0_1 Classes" Pacific J. Math 40(1972) 605-616
Pseudo jump operators: Jockusch, Shore " Pseudo jump operators I: the r.e. case" Trans. Amer. Math. Soc. 275 (1983) 599-609; " Pseudo-jump operators II: Transfinite iterations, hierarchies, and minimal covers" JSL 49 (1984) 1205-1236

Chapter 12

Pseudo-Jump Operators

Definition 12.0.1 *Pseudo-jump operators are defined for each index e: $J_e(A) = A \oplus W_e^A$. Such operators are called (1)-REA because the image is RE in A and above A. A 2-REA operator iterates this once (given two indices): $J_{e_2}(J_{e_1}(A))$. The ω-REA operators are each given by a recursive function f such that $J_f(A) = \oplus_{n\in\omega} J_{f\restriction n}(A)$. $J_{f\restriction n}(A)$ are n-REA operators. Note that if $f \restriction n$ gives the index of usual jump operator (iterated n times) then $J_f(A) \equiv_T A^{(\omega)}$.*

Theorem 12.0.2 (Completeness Theorem) *If J is an α-REA operator for $\alpha \leq \omega$ and $C \geq 0^{(\alpha)}$, then there is A such that $J(A) \equiv_T C \equiv_T A \vee 0^{(\alpha)}$.*

Proof. For $\alpha = 1$, $J = J_e(A) = A \oplus W_e^A$. We will build a tree $T_1 \leq 0'$, one of whose paths will be A. $T_1 \subseteq ID$, will be defined by recursion according to the following intuition.

- if τ is coded on n^{th} level of tree then $\tau \Vdash n \in J_e(A)$ or $\tau \Vdash n \notin J_e(A)$;
- labels on tree also code whether τ forced membership or non-membership.

Therefore, we define $T_1 : 2^{<\omega} \to (2^{<\omega} \times 2^{<\omega})$, $T_1(\sigma) = \langle \sigma_0, \sigma_1 \rangle$ such that

$$\sigma \subseteq \tau \implies T_1(\sigma)_i \subseteq T_1(\tau)_i \text{ for } i = 0, 1, \qquad \text{and} \qquad \sigma \mid \tau \implies T(\sigma)_0 \mid T(\tau)_0.$$

$T_1(\epsilon) = \langle \epsilon, \epsilon \rangle$. If we have $T_1(\sigma) = \langle \sigma_0, \sigma_1 \rangle$, $|\sigma| = n$ then define $T_1(\sigma\hat{\ } i)$ for $i = 0, 1$ by asking if there is $\tau \supseteq T(\sigma)_0$ such that $\tau \Vdash n \in J_e(A)$ a $\Sigma_1(A)$ question. If so, choose first such τ and let $T_1(\sigma\hat{\ } i) = \langle \tau\hat{\ } i, \sigma_1\hat{\ } 1 \rangle$. If there is no such extension, let $T_1(\sigma\hat{\ } i) = \langle \sigma_0\hat{\ } i, \sigma_1\hat{\ } 0 \rangle$.

Fix any $C \geq 0'$. Let $A = \cup_n \Pi_0(T_1(C \restriction n))$. By construction, $J_e(A) = \cup_n \Pi_1(T_1(C \restriction n))$. We now verify that A is as required for the theorem. That is, we want to check that $J_e(A) \equiv_T C \equiv_T A \vee 0'$.

- The definition of T_1 is recursive in $0'$, hence in C. Therefore, $A \leq_T C$ so $A \vee 0' \leq_T C$. Likewise, $J_e(A) \leq_T C$.

- We claim that $C \leq_T J_e(A)$. Suppose we want to compute $C(n)$. Let $\sigma = C \upharpoonright n$ (inductively). $J_e(A)$ can compute $T_1(C \upharpoonright n)$ because $J_e(A)(m)$ gives the answer to whether or not membership was forced at level m. Using these answers, the rest of the construction is recursive. Given $T_1(C \upharpoonright n)$, the construction asks if there is $\tau \supseteq T_1(C \upharpoonright n)_0$ such that $\tau \Vdash n \in J_e(A)$. But, $J_e(A)(n)$ can answer this question and so it knows which case the construction will be in. If the answer is yes, we can recursively look for an extension τ and then $T_1(C \upharpoonright (n)\hat{\ }i) = \langle \tau\hat{\ }i, \sigma_1\hat{\ }i\rangle$ so $C(n) = i$. But, A can tell us which way we branched on tree after τ and hence the value of i. Since $A \leq_T J_e(A)$, $J_e(A)$ can find i (which is $C(n)$). Thus, $C \leq_T J_e(A)$.

- We will be done once we show that $J_e(A) \leq_T A \vee 0'$. This is very similar to above, since we notice that $0'$ can answer the questions that $J_e(A)$ answered: e.g. is there finite extension which forces a Σ_1 question.

For general α, we will build a sequence of trees $T_\beta \leq 0^{(\beta)}$, $\beta \leq \alpha$ where $T_\beta(\sigma) = \langle \sigma_0, \dots \sigma_\beta\rangle$. Along the first coordinate, labels in the tree have to respect extension and incomparability but along all the other coordinates we only worry about extension. Moreover, the trees are nested in the sense that if $\langle \sigma_0, \dots, \sigma_\beta\rangle \in T_\beta$ and $\gamma < \beta$ then $\langle \sigma_0, \dots \sigma_\gamma\rangle \in T_\gamma$. We define the length of a sequence as the length of its last element. Think of n^{th} coordinate as coding $J_{f\upharpoonright n}(A)$. By induction on $\beta < \omega$ assume we have T_β. Define $T_{\beta+1}(\epsilon) = T_\beta(\epsilon)\hat{\ }\epsilon$. Then if we've defined $T_{\beta+1}(\sigma) = T_\beta(\tau)\hat{\ }\rho$ ($|\rho| = |\sigma| = m$), we want to define $T_{\beta+1}(\sigma\hat{\ }i)$. Ask if there is $\hat{\tau} \supseteq \tau$ such that $(T_\beta(\tau))_\beta \Vdash m \in J_{f(\beta)}(A)$. If so, let $T_{\beta+1}(\sigma\hat{\ }i) = T_\beta(\hat{\tau}\hat{\ }i)\hat{\ }(\rho\hat{\ }1)$. If not, let $T_{\beta+1}(\sigma\hat{\ }i) = T_\beta(\tau\hat{\ }i)\hat{\ }(\rho\hat{\ }0)$.

Start this procedure at nodes σ of length β in T_β and before that, copy T_β adding on ϵ as last coordinate. Then, can define T_ω as limit of T_β for $\beta < \omega$ and level n is fixed from T_n onwards so limit exists.

Let $C \geq_T 0^{(\alpha)}$. Define $A = \cup \Pi_0 T_\alpha(C \upharpoonright n)$. Claim that $J_{f\upharpoonright m}(A) = \cup_n \Pi_m T_\alpha(C \upharpoonright n)$ for each $m < \alpha$, and that $J_f(A) \equiv_T C \equiv A \vee 0^{(\alpha)}$. Note that if $\alpha < \omega$ then f is a finite function and that we only build trees up to T_α.

Proof: Showing that $J_{f\upharpoonright m}(A) = \cup_n \Pi_m T_\alpha(C \upharpoonright n)$ is straightforward by induction on m. $A \oplus J_{f(0)}(A)$ knows if $0 \in J_{f(0)}(A)$ and so can find $T_1(0\hat{\ }i)$ (by looking for extension on T_0, the identity tree). Then A can determine which of 0 and 1 the path follows because it is on the path and so gives us $C(0)$. Note that C could also have figure out how construction went because it knows value of $C(0)$. Moreover, $0'$ can also figure out construction because it knows if there is extension forcing the fact. What about $T_2(1\hat{\ }i)$? Construction asked for extension on T_1 which decides whether $0 \in J_{f(1)}(A)$. $J_{f(1)}(A)$ can answer this. Also, $0''$ can answer this question because $T_1 \leq_T 0'$ so this is a $0'(0') = 0''$ question. And, C knows the answer because if there is a second tree then $\alpha \geq 2$ and $C \geq 0''$. If the answer is no then the branching in the second tree is the same as the branching in the first tree. Since the second coordinate is the value of J_{e_0}, J_{e_0} can trace along what the path does in T_1 and thus figure out which way the branching goes. Then, it can calculate $C(1)$. If the answer is yes, then branch is first place along tree which makes it converge. But J_{e_0} can do tracing through path by comparing second coordinate with true value until find extension which is long enough to converge. To figure out branching in second tree, J_{e_0} must trace back branching of first tree. Thus, $C, 0^\alpha \oplus A, J_f(A)$ can each simultaneously compute the sequence of trees. ■

Corollary 12.0.3 *Every $C \geq_T 0^{(\omega)}$ is minimal cover. That is, there is $A < C$ such that C is minimal with respect to A.*

Proof. Relativizing the minimal degree argument from last time, for every A there is minimal cover M of A such that $M \leq_{wtt} A'$. The construction is uniform in A and the bound on the use of A' does not depend on A. Hence, there is an operator $\hat{J}(A) = \Phi_e^{A'}$ where the use of A' is bounded by recursive function f. For any such operator $\hat{J}$, there is ω-REA operator J such that $\hat{J}(A) \equiv_T J(A)$ for all A. Why? Because wtt reducibility is ω-r.e., $\hat{J}(A)(x) = \lim\limits_{s\to\infty} \Phi_e^A(x,s)$,$\Phi_e^A(x,0) = 0$, and the number of changes is bounded by a recursive function:

$$|\{s : \Phi_e^A(x,s) \neq \Phi_e^A(x,s+1)\}| \leq f(x).$$

We define the i^{th} column of $J(A)$ by

$$J(A)^{[i]} = \{\langle x,s\rangle : |\{t < s : \Phi_e^A(x,t) \neq \Phi_e^A(x,t+1)\}| = f(x) - i \;\&\; \Phi_e^A(x,s) \neq \hat{J}(A)\}$$

(Note that for $i = f(x)$ the column is empty.) Therefore, $J(A) \leq_T (f \oplus \hat{J}(A))$ so $J(A) \leq_T \hat{J}(A)$. Also,

$$x \in \hat{J}(A) \iff \langle f(x), x, 0\rangle \in J(A),$$

so $\hat{J}(A) \leq_T J(A)$ and $\hat{J}(A) \equiv_T J(A)$. Moreover, $J(A)^{[i+1]}$ is RE in $J(A)^{[i]}$ so $J(A)$ is an ω-REA operator because can take joins of columns. Hence $J(A)$ is an ω-REA operator Turing equivalent to $\hat{J}(A)$.

We apply the completeness theorem to J and C to get A such that $J(A) \equiv_T C$. Therefore, $\hat{J}(A) \equiv_T C$ and by assumption $\hat{J}(A)$ is a minimal cover of A. ■