Generalized High Degrees have the Complementation Property

Noam Greenberg, Antonio Montalbán and Richard A. Shore*
Department of Mathematics
Cornell University
Ithaca NY 14853

July 22, 2004

Abstract

We show that if $\mathbf{d} \in \mathbf{GH}_1$ then $\mathcal{D}(\leq \mathbf{d})$ has the complementation property, i.e. for all $\mathbf{a} < \mathbf{d}$ there is some $\mathbf{b} < \mathbf{d}$ such that $\mathbf{a} \wedge \mathbf{b} = \mathbf{0}$ and $\mathbf{a} \vee \mathbf{b} = \mathbf{d}$.

1 Introduction

A major theme in the investigation of the structure of the Turing degrees, $(\mathcal{D}, \leq_T)$, has been the relationship between the order theoretic properties of a degree and its complexity of definition in arithmetic as expressed by the Turing jump operator which embodies a single step in the hierarchy of quantification. For example, there is a long history of results showing that $\mathbf{0}'$ has many special order theoretic properties. To cite just a few: every countable partial order can be embedded below $\mathbf{0}'$ (Kleene and Post [KP54]); there are minimal degrees below $\mathbf{0}'$ (Sacks [Sac61]); $\mathbf{0}'$ cups to every degree above it (and so has the *cupping property*)(Friedberg [Fri57]); every degree below $\mathbf{0}'$ joins up to $\mathbf{0}'$ (and so has the *join property*)(Robinson [Rob72], Posner and Robinson [PR81]).

It was often hoped that some such property would distinguish either $\mathbf{0}'$ or some class of degrees closely related to it. For degrees below $\mathbf{0}'$, the notion of being close to $\mathbf{0}'$ (or $\mathbf{0}$ at the other end) was also measured by the jump operator via the high/low hierarchy: for $\mathbf{d} \leq_{\mathbf{T}} \mathbf{0}'$, $\mathbf{d} \in \mathbf{H}_n \Leftrightarrow \mathbf{d}^{(n)} = \mathbf{0}^{(n+1)}$ and $\mathbf{d} \in \mathbf{L}_n \Leftrightarrow \mathbf{d}^{(n)} = \mathbf{0}^{(n)}$. The questions then became at which levels of this hierarchy do the various properties of $\mathbf{0}'$ appear. The corresponding results for the above properties for $\mathbf{d} \leq_{\mathbf{T}} \mathbf{0}'$ are as follows: every countable partial order can be embedded below every $\mathbf{d} \notin \mathbf{L}_2$; there are minimal degrees below

*All the authors were partially supported by NSF Grant DMS-0100035.

every $\mathbf{d} \in \mathbf{H}_1$; every $\mathbf{d} \notin \mathbf{L}_2$ cups to every degree above it; every degree below $\mathbf{d}$ joins up to $\mathbf{d}$ for $\mathbf{d} \in \mathbf{H}_1$. (Except for the last, these results are all known to be sharp in terms of the high/low hierarchy. The sharpness of the first follows from the existence of a minimal degree $\mathbf{a} \in \mathbf{L}_2 - \mathbf{L}_1$: There is a minimal $\mathbf{a} < \mathbf{0}'$ not in $\mathbf{L}_1$ by Sasso [Sas74] and it is in $\mathbf{L}_2$ by Jockusch and Posner [JP78]. This minimal degree is easily seen to be weakly recursive in the sense of Ishmukhametov [Ish99] and then, by that work, has a strong minimal cover $\mathbf{b}$, i.e. $\{\mathbf{0}, \mathbf{a}, \mathbf{b}\}$ is an initial segment of the degrees. This proves the sharpness of the second fact. The existence of such an initial segment with $\mathbf{b}$ also below $\mathbf{0}'$ follows by Lerman's methods as outlined in [Ler83, XII.5.8]. The sharpness of the second is by Lerman [Ler86].) The techniques used to prove all these positive results are tied up with approximation methods, rates of growth conditions and domination properties. Thus they are of independent interest for relating degree theoretic properties of functions with such conditions.

In the setting of the degrees as a whole the analogous measure of the strength of a degree in terms of its jumps is the generalized high/low hierarchy: $\mathbf{d} \in \mathbf{GH}_n \Leftrightarrow \mathbf{d}^{(n)} = (\mathbf{d} \vee \mathbf{0}')^{(n)}$ and $\mathbf{d} \in \mathbf{GL}_n \Leftrightarrow \mathbf{d}^{(n)} = (\mathbf{d} \vee \mathbf{0}')^{(n-1)}$. (We take $\mathbf{GL}_0 = \{\mathbf{0}\}$.) All of the results mentioned above for $\mathbf{d} \leq_{\mathbf{T}} \mathbf{0}'$, for example, are true for all degrees as long as we use the generalized hierarchy. (Jockusch and Posner [JP78]; (Cooper [Coo73]) for $\mathbf{H}_1$ and Jockusch [Joc77] for $\mathbf{GH}_1$; Jockusch and Posner [JP78]; Posner [Pos77].) Once again approximations, rates of growth and domination properties play prominent roles in the constructions.

It was often hoped that these investigations would lead to a definition of $\mathbf{0}'$ in $\mathcal{D}$ or of some of the classes in the appropriate hierarchy in $\mathcal{D}$ or $\mathcal{D}(\leq \mathbf{0}')$. In fact, the jump operator has been defined in $\mathcal{D}$ by entirely different methods involving coding models of arithmetic and other arguments by Shore and Slaman [SS99] as have all of the high/low classes in $\mathcal{D}(\leq \mathbf{0}')$ with the exception of $\mathbf{L}_1$ by Nies, Shore and Slaman [NSS98]. The dream of a natural definition for any of these classes based on such order theoretic properties, however, still persists and can only be realized by investigations such as these. Moreover, the analysis of the relations between rates of growth and domination principles and the ordering of degrees remains intriguing on its own.

In this paper we consider the *complementation property* for degrees $\mathbf{d}$: for every $\mathbf{a} < \mathbf{d}$ there is a $\mathbf{b} < \mathbf{d}$ such that $\mathbf{a} \vee \mathbf{b} = \mathbf{d}$ and $\mathbf{a} \wedge \mathbf{b} = \mathbf{0}$. The primary result here is that $\mathbf{0}'$ has the complementation property. This result has a long history. R. W. Robinson [Rob72, cf. [PR81]] showed that every $\mathbf{a} \in \mathbf{L}_2$ has a complement in $\mathcal{D}(\leq \mathbf{0}')$; Posner [Pos77], that every $\mathbf{a} \in \mathbf{H}_1$ has one and Epstein [Eps75], that every r.e. $\mathbf{a}$ has one. Finally, Posner [Pos81] showed that every $\mathbf{a} \notin \mathbf{L}_2$ has a complement in $\mathcal{D}(\leq \mathbf{0}')$ and so $\mathbf{0}'$ has the complementation property. By a further argument using relativization and an additional division into cases along with some known results, Posner also showed that every $\mathbf{d} \in \mathbf{GH}_0$, i.e. $\mathbf{d} \geq \mathbf{0}'$, also has the complementation property. At the end of that paper, he posed four questions:

1. Is there a uniform proof of the complementation property for $\mathbf{0}'$ (i.e. eliminate the

split into cases and provide a single procedure that generates a complement for any nonrecursive $\mathbf{a} < \mathbf{0}'$)?

2. Does every nonrecursive $\mathbf{a} < \mathbf{0}'$ have a 1-generic complement?

3. Does every nonrecursive $\mathbf{a} < \mathbf{0}'$ have a complement of minimal degree?

4. Does every $\mathbf{d} \in \mathbf{GH}_1$ have the complementation property?

Slaman and Steel [SS89] answered questions 1 and 2 simultaneously by providing a uniform proof that every nonrecursive $\mathbf{a} < \mathbf{0}'$ has a 1-generic complement. Seetapun and Slaman circulated a sketch of a proposed affirmative answer to the third question around 1992 and that sketch has recently been extended and made into a proof by Lewis ([Lew03]). There have been several partial and related results about the fourth question. Epstein [Eps81], for example, showed that if $\mathbf{a} < \mathbf{h}$ are r.e. and $\mathbf{h} \in \mathbf{H}_1$ then $\mathbf{a}$ has a complement in $\mathcal{D}(\leq \mathbf{h})$. In this paper we supply a full affirmative answer:

Theorem 1.1. *Every degree $\mathbf{d} \in \mathbf{GH}_1$ has the complementation property, i.e. for every $\mathbf{a} < \mathbf{d}$ there is a $\mathbf{b} < \mathbf{d}$ such that $\mathbf{a} \vee \mathbf{b} = \mathbf{d}$ and $\mathbf{a} \wedge \mathbf{b} = \mathbf{0}$.*

Like the original proof that $\mathbf{0}'$ has the complementation property, our proof of this theorem is nonuniform. Rather than two case ($\mathbf{a} \in \mathbf{L}_2$ and $\mathbf{a} \notin \mathbf{L}_2$) we here have three separate cases, depending on the "distance" between $\mathbf{a}$ and $\mathbf{d}$. The three cases are:

1. $\mathbf{a} \in \mathbf{GL}_2$.

2. $\mathbf{a} \notin \mathbf{GL}_2$, but there is some function recursive in $\mathbf{d}$ which is not dominated by any function recursive in $\mathbf{a}$.

3. Every function recursive in $\mathbf{d}$ is dominated by some function recursive in $\mathbf{a}$.

Note that as defined these three cases are disjoint. The point is that if $\mathbf{a} \in \mathbf{GL}_2$ then $\mathbf{d}' = (\mathbf{d} \vee \mathbf{0}')' \geq (\mathbf{a} \vee \mathbf{0}')' = \mathbf{a}''$ and so there is a function recursive in $\mathbf{d}$ which dominates every function recursive in $\mathbf{a}$. They are also each nonempty. For (2) just take $\mathbf{d}' \geq \mathbf{a}''$. For (3) choose $\mathbf{a} \geq \mathbf{0}'$ and $\mathbf{d}$ any degree above $\mathbf{a}$ which is hyperimmune free with respect to $\mathbf{a}$ such as is given by relativizing either Miller and Martin [MM68] or the standard construction of a Spector minimal degree using total recursive binary trees to $\mathbf{a}$. (The point here is that the usual forcing argument, even if only attempting to force minimality, decides totality of every ϕ_e^A. Thus, if ϕ_e^A is total, A lies on a recursive binary tree T every path through which makes ϕ_e total. So, for every n there is a level s of the tree such that $\phi_e^{T(\sigma)} \downarrow$ for every σ at level s. This guarantees that ϕ_e^A is dominated by the recursive function which, given n, finds such an s and then outputs the maximum value of $\phi_e^{T(\sigma)}$ for σ on level s.) We should also note, however, that the proof we give for case (1) only needs $\mathbf{d}' \geq \mathbf{a}''$ and so would cover some pairs in cases (2) as well. The constructions of a complement for $\mathbf{a}$ in $\mathcal{D}(\leq \mathbf{d})$ in each of the three cases are quite distinct. We describe the intuitions and the motivations for each now and provide the precise constructions and verifications in the following sections.

Case 1: The existence of a complement for $\mathbf{a}$ below $\mathbf{d}$ in this case was already proved in his thesis by Posner [Pos77]. Like the proof that $\mathbf{L}_2$ degrees have complements in $\mathcal{D}(\leq \mathbf{0}')$ in Posner and Robinson [PR81] that proof was indirect (see [Ler83, IV.15]) and based on a nonuniform proof of the join theorem for $\mathbf{GH}_1$ (see [Ler83, IV.9]). It was also explicitly left open even in [Ler83] if the degree providing the join could be made $\mathbf{GL}_1$. This was answered by Lerman [Ler85] but again nonuniformly. We supply a uniform proof for this sharper version of the join theorem for $\mathbf{GH}_1$ which we then modify to get this case of the complementation theorem.

The basic idea of our proof of the join theorem for $\mathbf{d} \in \mathbf{GH}_1$ is, given $A <_T D \in \mathbf{d}$, to build a 1-generic set B in a construction recursive in D (so that $B <_T D$) that codes D and to have the choices we make at each stage depend on A, so that the construction itself (and hence D) can be recovered from $A \oplus B$. It combines the key idea of Jockusch and Shore's [JS84] proof of the Posner-Robinson join theorem for $\mathbf{0}'$ with Jockusch's [Joc77] use of the recursion theorem with degrees in $\mathbf{GH}_1$. This approach is maintained for the proof of the first case of the complementation theorem, except that to ensure that $A \wedge B \equiv_T 0$ we look for splits instead of strings forcing the jump. Since we are now looking for two strings rather than one, the coding is a little more complicated. Also, once we have found a split, we need the convergence of a functional with oracle A to let us know which string of the split we should take to diagonalize. This presents the problem that D cannot necessarily know if such a convergence will ever appear. However, by our case assumption, D can approximate whether a given functional with oracle A produces a total function (which is the only case we worry about). Thus, if we keep attacking some requirement, we will eventually believe the correct outcome and act appropriately.

Case 2: In this case, $\mathbf{a}$ is no longer in $\mathbf{GL}_2$, hence $\mathbf{d}$ cannot approximate questions regarding totality. However, as was noted by Jockusch and Posner [JP78] (using Martin's [Mar66] characterization of the domination properties of $\mathbf{H}_1$), this loss is offset by $\mathbf{a}$'s ability to approximate bounds for searches conducted by $\mathbf{0}'$: for every function $f \leq_T \mathbf{0}'$ (even $\mathbf{a} \vee \mathbf{0}'$) there is a function $g \leq_T \mathbf{a}$, not dominated by f.

We would like to imitate a construction of complements below $\mathbf{0}'$, and as in the previous case, we use the fact that $\mathbf{d}$ being generalized high enables us to approximate constructions which are naturally recursive in $\mathbf{0}'$. However, another special aspect of $\mathbf{0}'$ is that it is *recursively enumerable* . Dominating the modulus (computation) function for a recursively enumerable (even Δ_2) set enables us to compute the set itself. In other words, there is a function $d \leq_T \mathbf{0}'$ which isn't dominated by any function of lesser degree. In the standard constructions, this property is used to bound searches for facts which are r.e. in $\mathbf{a}$. As this seems to be an essential element of all such constructions, we simply assume (this is the second case) that we are given a function $d \leq_T \mathbf{d}$ which isn't dominated by any function recursive in $\mathbf{a}$, and carry out the construction (with the necessary modifications, of course). Given such a function we can adapt the uniform construction of complements below $\mathbf{0}'$ of [SS89] to our second case. If such a function does not exist, we find ourselves in the third case for which a completely different approach seems to be required.

Case 3: The key observation which enables us to construct complements in the third case is the following. Suppose that an A falling under the third case is given, and we manage to construct some $B \leq_T D$ such that $A \wedge B \equiv_T 0$ and such that $D \leq_T (A \oplus B)'$. In this case we must have $D \equiv_T A \oplus B$; if not, then we have

$$A \oplus B \ <_T \ D \leq_T (A \oplus B)',$$

in other words, $D \in \Delta_2(A \oplus B)$ but is not $\Delta_1(A \oplus B)$, which implies (see [Ler83, III.5.9]) that the computation function for D (as a set which is $\Delta_2(A \oplus B)$) is not dominated by any function recursive in $A \oplus B$, so also not by any function recursive in A. This contradicts the assumption that we are in case (3).

To construct B as required, we simply construct a set B of *minimal* degree, satisfying the requirement $D \leq_T (A \oplus B)'$. Since we can assume that $D \not\leq_T A'$ (by the same argument as above), we cannot have $B \leq_T A$. Since B is minimal, we get $A \wedge B \equiv_T 0$.

Now there is a mysterious aspect of our construction. We construct a set B *without reference* to the set A, and then use the assumed domination property of A to get our result. This implies that any B so constructed is automatically a complement for *every* A falling under the third case.

To accomplish this, we need to construct a B which will have properties which are fairly independent of the construction. We begin with a more controlled version of the construction of a minimal degree below a $\mathbf{d} \in \mathbf{GH}_1$ and then modify it to code in D in a way that can be recovered from B and any function g dominating a search function used in the construction. The case assumption for A will guarantee that there is such a function recursive in A.

We begin with the standard method of constructing (a Spector) minimal degree by forcing with perfect trees. One constructs a sequence of recursive perfect trees, each tree succeeded by a splitting subtree or an extension (full) subtree, forcing either that $B \leq_T \Phi(B)$ or that $\Phi(B)$ is recursive, respectively (of course, if total). To prove Sacks' [Sac61] result that there is a minimal degree below $\mathbf{0}'$, one introduces a priority argument version of this basic construction and uses partial recursive trees to get a minimal degree below $\mathbf{0}'$. The idea is to define the successor subtree as the splitting subtree, until we find an initial segment of the set we are building which does not split on our current tree, and then switch (permanently modulo action by higher priority requirements) to an extension (full) subtree.

The generalization of this technique to working below an arbitrary $\mathbf{GH}_1$ degree was introduced by Jockusch in [Joc77] where he proved that below every $\mathbf{GH}_1$ degree there is a minimal degree. Since, in this case, we do not have the power of $\mathbf{0}'$ to find whether a certain string splits on a tree or not, we keep looking for splits in hope of eventually finding one. The power of $\mathbf{GH}_1$, together with the recursion theorem, allows us, as in the previous cases, to guess, eventually correctly, whether such a search will be successful. We use this guess to determine what kind of tree the successor will be. Since we may guess incorrectly for some time, we may switch back and forth between splitting and extension (full) subtrees, but only finitely many times, and so we maintain the finite injury character of the construction.

In such a construction, the limit tree in a given place in the sequence of final trees is determined not only by the set B we construct and the question whether it splits or not on the previous tree, but also on the time during the construction at which we finally made the correct guess - this is because that stage determines what the *stem* of the tree will be. The primary difference between our construction and Jockusch's is that we eliminate this dependency by making the stem of the tree independent of the stage and dependent only on B and on the previous tree. As we shall see, this change allows us to recover the final list of trees from $(B \oplus g)'$, where g is any function dominating the search in the construction for splits when needed or a notification from our D recursive approximation to the $(D \vee 0')'$ question as to whether there is a initial segment of B above which there are no splits. We will finally use the fact that A falls under the third case to see that such a g can be obtained from A.

Our solution to the complementation problem for $\mathbf{GH}_1$ obviously leaves several natural open questions. Can the complements be taken to be either always 1-generic or always minimal? Is there a uniform procedure for computing a complement from D and $A = \{e\}^D$ for $0 <_T A <_T D \in \mathbf{GH}_1$?

Finally, before beginning the specific cases, we provide some general definitions. We fix a set $D \in \mathbf{d}$ and a set $A \in \mathbf{a}$ (later we may specify sets with particular properties). We construct a set B whose degree will be $\mathbf{a}$'s complement. The requirements we need to satisfy are:

- $\mathbf{C}$: D is coded in B so that $D \leq_T A \oplus B$.
- $\mathbf{Z}_\Phi$: If $\Phi(A) = \Phi(B)$ is total, then it is recursive.

(Here Φ varies over Turing functionals; we identify a functional, and the requirement associated with it, with its index.) We think of a Turing functional as a recursive function $\Phi : 2^{<\omega} \to 2^{<\omega}$ which is extendable continuously to a function $\Phi : 2^\omega \to 2^{\leq\omega}$. If $q \in 2^{<\omega}$ and $n \in \mathrm{dom}\ \Phi(q)$ we sometimes write $\Phi(q; n)\downarrow$ and write $\Phi(q; n) = \Phi(q)(n)$.

We fix an implicit recursive bijection $\omega \leftrightarrow \omega^{<\omega}$. We let, for $n < \omega$,

$$\langle (n)_0, (n)_1, \ldots, (n)_{\ell(n)-1} \rangle$$

be the sequence associated with n.

In this paper, *tree* means a function tree, that is a partial function $T : 2^{<\omega} \to 2^{<\omega}$ which preserves order and nonorder and whose domain is closed under initial segments.

Suppose that Φ is a functional. A Φ-*split* is a pair of strings q_0, q_1 such that $\Phi(q_0) \perp \Phi(q_1)$, i.e. for some n, $\Phi(q_0; n) \downarrow \neq \Phi(q_1; n) \downarrow$. If (q_0, q_1) is a Φ-split, then we let $n_\Phi(q_0, q_1)$ be the least n witnessing this fact.

A split (q_0, q_1) *extends* p if both q_0 and q_1 extend p. If there is some Φ-split extending p, we say that p Φ-*splits* . If p Φ-splits, then we let $(q^0_\Phi(p), q^1_\Phi(p))$ be the least split extending p, and let $n_\Phi(p) = n_\Phi(q^0_\Phi(p), q^1_\Phi(p))$.

We say that a function $f : \omega \to \omega$ *dominates* the function $g : \omega \to \omega$ (and write $g <_* f$) if for all but finitely many n, $g(n) < f(n)$.

We use $\subset$ to denote containment whether proper or not.

2 Case One

We begin with a uniform proof of the join property (via a 1-generic) for $\mathbf{d} \in \mathbf{GH}_1$ that we then modify to produce a complement when A falls under Case 1.

Theorem 2.1. *If* $0 <_T A <_T D \in \mathbf{GH}_1$ *then there is a uniform procedure producing a 1-generic* B *such that* $A \vee B \equiv_T D$.

Proof. In addition to the coding the requirement $\mathbf{C}$, we here have just those for 1-genericity:

- $\mathbf{N}_\Phi$: $\Phi(B) \downarrow$ or $(\exists n)(\forall q \supseteq B \restriction n)(\Phi(q) \uparrow)$.

(Here we diverge from the usage described above and assume that the Turing functionals Φ take no input).

Without loss of generality (or uniformity), we may assume that A is the set of (natural number) codes for the initial segments of a set $\tilde{A}$ of the same degree. The set B will be constructed as an increasing union of strings $\langle \beta_s \rangle$. However, for the sake of the applicability of the recursion theorem, we also present B as a finer union of initial segments $\langle p_t \rangle_{t<\omega}$. The construction is recursive in D, hence the complexity of the question

$$\exists t\, (p_t \text{ has no extension } q \text{ such that } \Phi(q)\downarrow) \ ?$$

is $(D \vee 0')' \equiv_T D'$. Thus its answer can be approximated recursively in D. By the recursion theorem, we assume that we have a function $G \leq_T D$ such that the limit $\lim_s G(\Phi, s)$ is the answer to the question. At stage s of the construction we proceed as follows:

Let $p_1 = \beta_1 = \langle\rangle$. We are given $\beta_s = p_t$ for some t and we work for requirement $\mathbf{N}_\Phi$ for $\Phi = (s)_0$.

At substage k of stage s, we let $p_{t+k} = p_t\widehat{\ }0^k$. We find the least $r_0 > t + k$ such that either $G(\Phi, r_0) = \texttt{yes}$ or a q extending p_{t+k} with $\Phi(q)\downarrow$ is found after r_0 many steps (this search halts, for if there is no such q, then p_{t+k} is a witness to $\lim G(\Phi, r) = \texttt{yes}$).

Now if we didn't find a q as requested, we let l be the least element of A greater than k (of course A is infinite), and let $\beta_{s+1} = p_{t+k+1} = p_t\widehat{\ }0^l\frown 1\frown D(s)$.

Otherwise, we take the least $q \supset p_{t+k}$ such that $\Phi(q)\downarrow$ and such that for some l, $p_t\frown 0^l\frown 1 \subset q$ (the string we find can always be extended to such a string). If $l \in A$ then we do nothing and move to the next substage. Otherwise, we let $\beta_{s+1} = p_{t+k+1} = q\frown D(s)$.

To verify that this construction produces the desired B, we first need to show that there is no final stage s, in other words, that at every stage we eventually define β_{s+1} and move to the next stage. Assume the opposite holds and that s is the last stage. Thus at every substage k of s, we find the least q extending $\beta_s\widehat{\ }0^k$ containing at least one additional 1 such that $\Phi(q)\downarrow$ and this q extends $\beta_s\widehat{\ }0^l\frown 1$ for some $l \in A$. Thus we can recursively enumerate infinitely such l and so an infinite subset of A. As this would compute A recursively we have the desired contradiction.

To see that B is 1-generic consider any Φ. Find a stage s late enough such that the guess for $\mathbf{N}_\Phi$ is correct at stage s and $(s)_0 = \Phi$. If $G(\Phi) = \texttt{yes}$ then there is an initial segment of B with no extension q such that $\Phi(q)\downarrow$. If not, then we defined β_{s+1} such that $\Phi(\beta_{s+1})\downarrow$.

Finally to see that $D \leq_T A \oplus B$ it is clearly enough to show that $\langle\beta_s\rangle \leq_T A \oplus B$: Given β_s, we find the k such that $\beta_s\widehat{\ }0^k\frown 1 \subset B$. If $k \in A$ then $|\beta_{s+1}| = |\beta_s| + k + 2$. Otherwise, we can recursively find the least q extending $\beta_s\widehat{\ }0^k\frown 1$ such that $\Phi(q)\downarrow$. In this case $|\beta_{s+1}| = |q| + 1$. □

We now turn to case 1 of the complementation theorem.

2.1 Construction

The situation is as in the proof of the join theorem above except that we have requirements $\mathbf{Z}_\Phi$ in place of $\mathbf{N}_\Phi$ and so our function G approximates the answer to the question

$$\exists t\ \big(p_t \text{ does not } \Phi\text{-split}\ \big)\ ?$$

Also, $A'' \leq_T D'$ so we can approximate $\mathrm{Tot}(A)$ recursively in D: we have a function $T \leq_T D$ such that for all functionals Φ, $\lim_s T(\Phi, s) = \texttt{total}$ if $\Phi(A)$ is total, and $\lim_s T(\Phi, s) = \texttt{nottotal}$ otherwise.

Construction of B. Let $p_1 = \beta_1 = \langle\rangle$. At stage s, we are given $\beta_s = p_t$ for some t and we work for requirement $\mathbf{Z}_\Phi$ for $\Phi = (s)_0$.

At substage k of stage s, we let $p_{t+k} = p_t\widehat{\ }0^k$. We find the least $r_0 > s + k$ such that either $G(\Phi, r_0) = \texttt{yes}$ or a Φ-split extending p_{t+k} is found (the least one) after

r_0 many steps (this search halts, for if there is no such split, then p_{t+k} is a witness to $\lim G(\Phi, r) = \texttt{yes}$). In the latter case, we find the least $r_1 > r_0$ such that either $T(\Phi, r_1) =$ $\texttt{not total}$ or $\Phi(A; n_\Phi(p_{t+k})) \downarrow [r_1]$ (again, the search halts because no convergence witnesses $\lim T(\Phi, r) = \texttt{not total}$).

Now if we didn't find a split, or didn't find convergence, we let l be the least element of A greater than k (of course A is infinite), and let $\beta_{s+1} = p_{t+k+1} = p_t^\frown 0^{l\frown} 1^\frown D(s)$.

Otherwise, we know which element of the split q is the one which gives the wrong answer about $\Phi(A; n_\Phi(p_{t+k}))$; we may assume it extends some $p_t^\frown 0^{l\frown} 1$ for some $l \geq k$. If $l \in A$ then we do nothing and move to the next substage. Otherwise, we let $\beta_{s+1} = p_{t+k+1} = q'^\frown D(s)$, where q' is the least string $\subseteq q$ and extending $p_t^\frown 0^{l\frown} 1$ such that $\Phi(q') \perp \Phi(A)$. $\Diamond$

2.2 Verifications

We first need to show that there is no final stage s, in other words, that at every stage we eventually define β_{s+1} and move to the next stage. Assume the opposite holds and that s is the last stage. Thus at every substage k of s, the least Φ-split (q_k^0, q_k^1) extending $\beta_s^\frown 0^k$ is found, and we know that at least one of q_k^0, q_k^1 extends $\beta_s^\frown 0^l 1$ for some $l \in A$.

Thus, we can recursively enumerate infinitely many pairs (i_k, j_k) (where both coordinates get larger and larger; $i_k, j_k \geq k$) such that at least one of i_k, j_k is in A.

Now we can compute A and get a contradiction. Recall that the elements of A are codes for initial segments of $\tilde{A}$. Let σ_k be the string coded by i_k and τ_k the string coded by j_k.

Now there are two cases. Suppose that for all $\sigma \subset \tilde{A}$ there is some k such that $\sigma \subset \sigma_k, \tau_k$. Then we can compute $\tilde{A}$ by enumerating

$$\{\sigma_k \cap \tau_k \,:\, k < \omega\}.$$

Otherwise, there is some $\sigma^* \subset \tilde{A}$ and some k^* such that for all $k > k^*$, σ^* is extended by exactly one of σ_k, τ_k, and this one has to be an initial segment of $\tilde{A}$. In this case we can compute $\tilde{A}$ by enumerating

$$\{\sigma_k \,:\, k > k^*, \sigma^* \subset \sigma_k\} \cup \{\tau_k \,:\, k > k^*, \sigma^* \subset \tau_k\}.$$

Lemma 2.2. *$A \wedge B \equiv_T 0$.*

Proof. Fix Φ such that $\Phi(A)$ is total. Find a stage s late enough such that the guesses for $T(\Phi)$ and $G(\Phi)$ are correct at stage s and $(s)_0 = \Phi$. If $G(\Phi) = \texttt{yes}$ then $\Phi(B)$, if total, is recursive. If not, then we must have diagonalized at stage s, so that $\Phi(A) \neq \Phi(B)$. $\square$

Lemma 2.3. *$\langle \beta_s \rangle \leq_T A \oplus B$.*

Proof. Given β_s, we find the k such that $\beta_s{}^\frown 0^k{}^\frown 1 \subset B$. If $k \in A$ then $|\beta_{s+1}| = |\beta_s| + k + 2$. Otherwise, from A we can find the q such that $\beta_{s+1} = q^\frown D(s)$. $\square$

Corollary 2.4. $D \leq_T A \oplus B$.

3 Case Two

The construction takes place on a tree T; B will be a branch on this tree. In Slaman and Steel's construction, a requirement $\mathbf{Z}_\Phi$ strives to find a Φ-split on T, and tries to let B extend that part of the split which gives the wrong answer about $\Phi(A)$. Since about $\mathrm{Tot}(A)$ we know naught, the strategy is to find such a split and then prevent weaker requirements from extending B beyond this split, until $\mathbf{Z}_\Phi$ makes up its mind (due to a convergence of $\Phi(A)$) or decides to give up. In the original construction, the requirement gives up at the next stage it receives attention, at which it is either guaranteed a string which doesn't split on T (a *negative* win), or it is presented with a new split unto which its fortunes are now entrusted.

It is the domination property of d which is used to show that each requirement cannot be foiled for ever.

The first difficulty we come across is that in the construction of the tree, we wish to find out whether a certain node can be extended by some split. Instead of asking $\mathbf{0}'$, we construct the tree (ignoring the coding requirement for a while) recursively in A by looking, at level s, for splits, for $g(s)$ many steps, where $g \leq_T A$ is a function not dominated by some fixed $\tilde{f} \leq \mathbf{0}'$ which bounds the search needed to find all splits which exist. Thus in many levels devoted to some $\mathbf{Z}_\Phi$, the requirement falsely believes that no splits exist (and thus that it needs no action to succeed); but the domination properties of g ensure that, if there are infinitely many splits, splits will be found infinitely often, giving the requirement ample breathing space to act.

These changes create two new difficulties. As described earlier, a requirement $\mathbf{Z}_\Phi$ imposes its will on weaker requirements (which threaten to extend B beyond a split, while $\mathbf{Z}_\Phi$ is still waiting for convergence of $\Phi(A)$). This restraint is built into the construction of the tree, and is only imposed (for each split) for a single stage only, since at the next stage $\mathbf{Z}_\Phi$ looks ahead to find either a new split or no splits, in which case it is satisfied anyway. In our case, $\mathbf{Z}_\Phi$ does not know how long it needs to wait, thus restraints need to be made explicit. Fairness is maintained by observing that at the "true stages" (those stages at which $g > f$, meaning that the answers for the question "is there a split?" are correct) all unnecessary restraints are dropped.

However, each requirement needs to determine for *itself* whether to pursue positive satisfaction (by waiting for the next split) or to give up and let B extend a node beyond which no splits are found yet, hoping that indeed no splits will appear later. It turns out that D is sufficiently strong to enable $\mathbf{Z}_\Phi$ to guess which course to take (using the recursion theorem, of course). This solves this second problem.

We now give the details of the construction. For simplicity, we adopt a modular approach and ignore the coding requirement **C** for now, and later describe how to modify the construction to satisfy this requirement.

3.1 Construction of the Tree

We first define the function f which bounds the search for splits. Let

$$f_0(s) = \max\{\langle q^0_\Phi(p), q^1_\Phi(p)\rangle : p \in 2^{\leq s},\ \Phi \leq sp\ \Phi\text{-splits}\}.$$

Next, define f by recursion: $f(0) = f_0(0)$; $f(s+1) = f_0(f(s)+1)$. Finally, let $\tilde{f}(s) = f(\langle s, s\rangle)$. Observe that $\tilde{f}$ is recursive in $\mathbf{0}'$.

To level k of the tree we attach requirement $\mathbf{Z}_\Phi = (k)_0$. k is called a Φ-*level*.

We find some $g \leq_T A$, not dominated by $\tilde{f}$. We use g to define our tree T. $T(\langle\rangle) = \langle\rangle$. If $T(\sigma)$ is defined and equals p, and $|\sigma|$ is a Φ-level, then we look for a Φ-split extending p which appears before stage $g(s)$. If one is found, then we define $T(\sigma\widehat{\ }i) = q^i_\Phi(p)$. Otherwise, we let $T(\sigma\widehat{\ }0) = p\widehat{\ }0$ and leave $T(\sigma\widehat{\ }1)$ undefined.

We say that a string p is a *node* if it is in range T. We let

$$L(k) \ = \ \{T(\sigma) : |\sigma| = k\},$$

and also write level$(p) = k$ for all $p \in L(k)$.

It is easily verified that T is $\Delta_1(A)$ (so not only is T partial recursive in A but also dom T is recursive in A). We are also interested in verifying how closely this tree reflects the truth (about existence of splits).

Lemma 3.1. *For all k and $p \in L(k)$, $|p| < f(k)$.*

Proof. This is true by induction; the inductive step holds because the elements of the $k+1^{st}$ level are least splits (or immediate successors) of strings of level k, and f bounds all such splits. □

Definition. A level k is called Φ-*true* if it is a Φ-level, and $g(k) > f(k)$.

Thus if k is a Φ-true level then for all $p \in L(k)$, if there is no Φ-split on T extending p then indeed there is no such split at all.

Lemma 3.2. *For every Φ there are infinitely many Φ-true levels.*

Proof. We know that there are infinitely many $n \geq \Phi$ such that $g(n) \geq \tilde{f}(n)$. For each such n, if we let $k = \langle\Phi, n\rangle$, then

$$g(k) = g(\langle\Phi, n\rangle) \geq g(n) \geq f(\langle n, n\rangle) \geq f(\langle\Phi, n\rangle) = f(k).$$

□

3.2 The Construction

We now describe the construction of a set $B <_T D$ such that $A \wedge B \equiv_T 0$. We construct B as the union of an increasing sequence of initial segments $\langle \beta_s \rangle$ which are on T.

At stage s, each requirement $\mathbf{Z}_\Phi$ may impose a restraint $r(\Phi)[s]$. This is a level k of the tree (beyond the level of β_s), and the aim of the restraint is to prevent weaker requirements from extending β_{s+1} beyond the k^{th} level of T. If a stronger requirement $\mathbf{Z}_\Psi$ extends β_{s+1} beyond the k^{th} level of T then we say that it *injures* $\mathbf{Z}_\Phi$. We let

$$R(\Phi)[s] \;=\; \min\left(\{r(\Psi)[s] \,:\, \Psi < \Phi r(\Psi)[s]{\downarrow}\} \cup \{s\}\right).$$

This is the restraint imposed on requirement $\mathbf{Z}_\Phi$ at stage s. Thus at s, $\mathbf{Z}_\Phi$ is prohibited from defining β_{s+1} beyond the $R(\Phi)^{th}$ level of T.

There are various states in which a requirement $\mathbf{Z}_\Phi$ may find itself at a Φ-stage s. It may be *positively satisfied* at s if we have already forced that $\Phi(B) \perp \Phi(A)$, i.e. for some n we have

$$\Phi(A;n)\downarrow \neq \Phi(B;n)\downarrow [s].$$

If $\mathbf{Z}_\Phi$ is positively satisfied at some stage then it is satisfied until the end of time and doesn't need to act ever again.

We wish to describe when $\mathbf{Z}_\Phi$ is satisfied *negatively*. As explained above, this requirement will feel satisfied if it can make B extend some node p which doesn't Φ-split. Now without the power of $\mathbf{0}'$, this kind of satisfaction may be temporary, for the requirement may place its bets on some such p only to discover later some Φ-split extending p.

To make its decision whether to believe some negative satisfaction (or keep searching for splits), at various stages $\mathbf{Z}_\Phi$ will put some strings on a "testing list" ℓ_Φ (which can be thought of as a partial function defined by $\mathbf{Z}_\Phi$ during the construction). This will be the list of strings that $\mathbf{Z}_\Phi$ will test, to see if they may provide negative satisfaction.

We thus say that $\mathbf{Z}_\Phi$ is *negatively satisfied* at s if no Φ-splits extending the last element put on ℓ_Φ are found in $g(s)$ many steps.

[We note, that if $\mathbf{Z}_\Phi$ is negatively satisfied at a true stage s, then this satisfaction is correct and permanent; at true stages we find all possible splits above nodes on levels $\leq s$ on the tree.]

Now to decide whether it should even attempt to believe that negative satisfaction could come from some node p, $\mathbf{Z}_\Phi$ will try to find an answer to the question

> Is there some node p which is put on ℓ_Φ and doesn't split?

This question can be answered by $(D \oplus 0')' \equiv_T D'$. So, by the recursion theorem and the limit lemma, there is a function $G \leq_T D$, that we can use during the construction, which approximates the answer (i.e. $\lim_s G(\Phi, s) = \texttt{yes}$ if the answer to the question is yes, and $\lim_s G(\Phi, s) = \texttt{no}$ otherwise.)

Let

$$L(k, p) \;=\; \{T(\sigma) : |\sigma| = k \text{ and } \mathrm{T}(\sigma) \supseteq \mathrm{p}\},$$

this is the collection of nodes of the k^{th} level of T which extend p. At stage s, let $L(k, s) \;=\; L(k, \beta_s)$.

We say that $p = T(\sigma)$ *splits on* T if both $T(\sigma\widehat{\ }0)$ and $T(\sigma\widehat{\ }1)$ are defined, i.e. if a relevant split exists and was discovered by step $g(|\sigma|)$ and put on T.

Construction of B, with oracle D. At stage 0, we let $\beta_1 = \langle\rangle$, we set the lists ℓ_Φ to be empty, and let $r(\Phi)$ be undefined for all Φ.

If Φ is initialized (at any stage), then $r(\Phi)$ is removed.

Stage s: Suppose that s is a Φ-stage.

If $\mathbf{Z}_\Phi$ believes it is satisfied (positively or negatively), then we skip this stage; $r(\Phi)$ is not defined.

Otherwise, $\mathbf{Z}_\Phi$ may attempt to act, starting ambitiously.

Positive action

If $\mathbf{Z}_\Phi$ can legally extend β_s to be permanently satisfied, it does so. Namely, if there is some p on T, extending β_s, on a level below $R(\Phi)[s]$, such that for some n we have $\Phi(p; n) \neq \Phi(A; n)[d(s)]$ then $\mathbf{Z}_\Phi$ lets $\beta_{s+1} = p$. $r(\Phi)$ is removed (as the requirement is satisfied). [d is the function recursive in D, not dominated by any function recursive in A.]

Negative attempts

Otherwise, we look for some p of level beneath $R(\Phi)[s]$ (but which extends β_s) which is not yet on ℓ_Φ and which is not yet seen to Φ-split. If such a p exists, we add the first one to ℓ_Φ. We now perform a *test* to see if we believe p can be a witness for negative success: we find the least $t > s$ such that either $G(\Phi, t) = \texttt{yes}$, or some Φ-split extending p is found by stage t. This search has to halt since if there is no split to be found, then p is a witness to the limit of $G(\Phi, t)$ being $\texttt{yes}$. If a Φ-split was found then we may try with a new node p (until we run out). Otherwise, $\mathbf{Z}_\Phi$ sets $\beta_{s+1} = p$, and removes $r(\Phi)$.

If we run out of p's without acting, the requirement does nothing at this stage, but we calculate

Restraint

If $r(\Phi)$ is not defined, then it is set to be level(β_s). Otherwise, if there is some Φ-level k such that $r(\Phi) < k \leq R(\Phi)[s]$ and such that *every* $p \in L(k,s)$ splits on T, then we update $r(\Phi)$ to be the maximal such k. If there is no such k, then $r(\Phi)$ is left unaltered.

Whenever $\mathbf{Z}_\Phi$ acts and extends β_s, it initializes all weaker requirements. $\Diamond$

3.3 Verifications

Since this is an oracle construction, we have $B \leq_T D$. We first check that the construction is fair.

Lemma 3.3. *Every requirement is initialized only finitely many times, and eventually stops acting itself.*

Proof. Since initialization occurs only by action of a stronger requirement, we assume by induction that after $\hat{s}$, no requirement stronger than $\mathbf{Z}_\Phi$ acts again and so $\mathbf{Z}_\Phi$ is never initialized again.

If at any time $\mathbf{Z}_\Phi$ acts positively, then its satisfaction is permanent and it will never act again. We now consider what happens to $\mathbf{Z}_\Phi$ after stage $\hat{s}$.

Infinite negative action is also impossible. For either $\lim G(\Phi, s) = \texttt{no}$, in which case after some stage we will never believe some string is a likely candidate for negative satisfaction; or $\lim G(\Phi, s) = \texttt{yes}$, by virtue of a witness p which was put on ℓ_Φ; when we put p on the list, we performed the test and acted for p (because it really doesn't split); and negative satisfaction is permanent from then on (no splits are ever discovered, and no new strings are put on the list). $\square$

We digress to prove that the requirements succeed if they act.

Lemma 3.4. *If* $\mathbf{Z}_\Phi$ *ever acts positively, then it is met.*

Proof. Because then $\Phi(A) \neq \Phi(B)$. $\square$

Lemma 3.5. *If* $\mathbf{Z}_\Phi$ *is eventually permanently satisfied negatively by some* p *then it is met.*

Proof. We have some $p \subset B$ such that no Φ-splits extend p. Then $\Phi(B)$, if total, is recursive. $\square$

Now if a requirement is not eventually satisfied as in the last two lemmas, then there is a last stage at which the requirement acts, necessarily a negative action. Since this does not lead to permanent satisfaction, at some later stage the last action taken is discovered to be insufficient (a split is discovered, extending the last element of ℓ_Φ); after that stage, since the requirement never acts again, it also never believes it is satisfied. In this case we say that $\mathbf{Z}$ is *eventually unsatisfied.*

We let $s^*(\Phi)$ be the least Φ-stage after which neither $\mathbf{Z}_\Phi$ nor any stronger requirement ever acts again, and after which $\mathbf{Z}_\Phi$ is either permanently satisfied or eventually unsatisfied.

A requirement may also disturb weaker requirements by imposing restraint, so to check fairness, we need to verify that this restraint is not too prohibitive. If a requirement is permanently satisfied (positively or negatively) then it stops imposing any restraint.

Otherwise we have

Lemma 3.6. *Suppose that $\mathbf{Z}_\Phi$ is not eventually satisfied. Then at $s^*(\Phi)$, $r(\Phi)$ is defined; thereafter it is never removed and cannot decrease. For all $s \geq s^*(\Phi)$, $\mathrm{level}(\beta_s) \leq r(\Phi)[s]$.*

Proof. The first part is because $\mathbf{Z}_\Phi$ is never initialized again, or acts again. The second follows because weaker requirements respect $r(\Phi)$. □

We need to examine what happens to the restraint if a certain requirement is eventually unsatisfied. For this we use the true levels.

Lemma 3.7. *Suppose that $\mathbf{Z}_\Phi$ is eventually unsatisfied, $k > s^*(\Phi)$ is a Φ-true level and that $s > k$ is a Φ-stage such that $\mathrm{level}(\beta_s) \leq k \leq R(\Phi)[s]$. Then at s, $r(\Phi)$ is increased to at least k.*

Proof. If not, then we must have some $p \in L(k, s)$ which does not split on T. But k is a true stage; thus p doesn't split, and $\mathbf{Z}_\Phi$ would act negatively at stage s. □

We get

Lemma 3.8. $\lim_s R(\Phi)[s] = \infty$.

Proof. Assume (by induction) that $\lim R(\Phi)[s] = \infty$. Suppose, for contradiction, that $\lim r(\Phi, s) = r$ is finite. We can choose some Φ-true level $k > r$ and wait for a later Φ-stage $s > s^*(\Phi)$ such that $R(\Phi)[s] > k$. $\mathrm{level}(\beta_s) \leq r < k$; then we must have $r(\Phi)[s] \geq k$, contradiction. □

Lemma 3.9. $\lim_s |\beta_s| = \infty$.

Proof. There are infinitely many Φ such that, for all X, $\Phi(X) = \langle\rangle$, so there are no Φ-splits. For all these Φ, $\mathbf{Z}_\Phi$ will act negatively the first time it receives attention. Therefore, β_s is extended infinitely often. □

All that is left is to verify that $\mathbf{Z}_\Phi$ is met, even if it is eventually unsatisfied.

Lemma 3.10. *If $\mathbf{Z}_\Phi$ is eventually unsatisfied, then $\Phi(A)$ is not total.*

Proof. Suppose otherwise, and let $h(n)$ be the stage at which $\Phi(A;n)\downarrow$; $h \leq_T A$. We show that h dominates d for a contradiction.

Let $s_1, s_2, \ldots$ be the stages, after $s^*(\Phi)$, when $r(\Phi)[s]$ is increased. Let $r_i = r(\Phi)[s_i]$, and let p_i be the element of the r_i^{th} level of T which is an initial segment of B. Since $p_i \in L(r_i, s_i)$ ($\beta_{s_i} \subset B$), by the instructions of the construction, we must have that p_i splits on T. We let n_i be the splitting point, i.e. $n_i = n_\Phi(p_i)$.

We claim that for all $i < \omega$, $h(n_i) \geq d(n_{i+1})$. At stage s_{i+1}, $\mathbf{Z}_\Phi$ did not act positively. However, the Φ-split of p_i was still available for $\mathbf{Z}_\Phi$ to pick for positive satisfaction; the restraint until s_{i+1} was r_i and $r_i \leq R(\Phi, s_i) \leq R(\Phi, s_{i+1})$ (the latter inequality holds because after s_i no requirement stronger than $\mathbf{Z}_\Phi$ acts). This implies that $\Phi(A;n_i)\uparrow [d(s_{i+1})]$, in other words,

$$d(s_{i+1}) < h(n_i).$$

Of course, $n_i < s_i$ so $d(n_{i+1}) < d(s_{i+1}) < h(n_i)$.

Both h and d are increasing functions, so the last inequality implies that h dominates d: for every $n > n_1$, there exits i such that $n_i < n \leq n_{i+1}$. Then

$$d(n) \leq d(n_{i+1}) < h(n_i) \leq h(n).$$

$\square$

3.4 Coding D into B

In this section we show how to modify the construction to satisfy the coding requirement $\mathbf{C}$; this will ensure that indeed B is a complement for A below D.

To do this, we specify an infinite, recursive set of levels on f (say all k such that $(k)_0 = 0$) which will be devoted to $\mathbf{C}$ (by the padding lemma this does not disturb the previous construction). Let $\langle n_i \rangle_{i<\omega}$ be an increasing enumeration of this set. We define a tree $T_D \subset T$ (as functions) by removing all nodes extending $T(\sigma\hat{\ }(1 - D(i)))$ for $\sigma \in 2^{n_i}$. Namely,

$$T_D \;=\; T \restriction \{\sigma \,:\, \forall n_i < \operatorname{dom}\sigma \,:\, \sigma(n_i) = D(i)\}.$$

Lemma 3.11. *$T_D \leq_T D$ and for every $X \in [T_D]$, $D \leq_T X \oplus A$.*

Proof. To find whether $i \in D$, A can find (on T) the unique σ on level $n_i + 1$ such that $T(\sigma) \subset X$. Since $X \in [T_D]$ we have $\sigma(n_i) = D(i)$. $\square$

Thus all we need to do is repeat the previous construction, using T_D instead of T. This can be done because the construction is recursive in D; the construction follows through verbatim.

4 Case Three

4.1 A minimal degree below D

We first give our version of the construction of a minimal degree below D. As in Sacks's and Jockusch's constructions, we will construct a sequence of partial recursive trees $\langle T_k \rangle$ and an increasing sequence of strings β_s whose union B is a path on all of the trees. Later we will show how to modify this construction so that D can be recovered from $\langle T_k \rangle$. Now we just build B such that $\langle T_k \rangle \leq_T (A \oplus B)'$.

We follow Lerman's [Ler83] definitions of splitting subtrees. If T is a tree and Φ is a functional, then a Φ-splitting subtree of T is a subtree S such that for all $p \in \mathrm{dom}\ T$, $T(p^\frown 0)$ and $T(p^\frown 1)$, if defined, are a Φ-split. The canonical splitting subtree above a node $T(p)$ (denoted $S = Sp(T, \Phi, p)$) is defined by induction, letting the stem be $T(p)$; given $S(q)$ on T, $S(q^\frown 0), S(q^\frown 1)$ is defined to be the least Φ-split on T extending $S(q)$, and undefined if none is found.

We say that $X \in [T]$ *Φ-splits* on T if for all $\sigma \subset X$, there is a Φ-split on T extending σ.

The requirement to be satisfied is

$\mathbf{M}_\Phi$ If $B\Phi$-splits on every tree T_k then it lies on a Φ-splitting tree.

Associating requirements with numbers as usual, the strategy to satisfy $k = \mathbf{M}_\Phi$ is to let $T_{k+1} = T_k$ if B does not Φ-split on T_k, and $Sp(T_k, \Phi, p)$ for some p otherwise. We make p depend on B directly, by letting p be the *shortest string* such that $B \in Sp(T_k, \Phi, p)$.

As usual, the construction will have approximations $T_k[s]$ to T_k and an increasing sequence $\langle \beta_s \rangle$ of strings whose union is B. During each stage of the construction the trees $T_k[s]$ are enumerated a certain number of steps to find splits above β_s. We will make sure that once T_k has stabilized, if B splits on T_k then at every s there is a split found on T_k extending β_s. Thus from the construction we can calculate a function f which bounds the search needed to find splits on the trees. Namely, we have a function $f \leq_T D$ such that for all $k = \mathbf{M}_\Phi$, if B Φ-splits on T_k then for almost all n, a Φ-split extending $B \upharpoonright n$ will be found on T_k after enumerating the tree for $f(n)$ many steps.

[We note that it is not necessary to assume that $\mathrm{dom}\, \beta_s \geq s$ to get such a function; to calculate $f(n)$ we can simply wait for a stage at which $\mathrm{dom}\, \beta_s > n$. This will be helpful in the full construction we give at the end].

First, we would like to ensure that the above conditions on the construction indeed produce the desired result. Suppose, then, that B, $\langle T_k \rangle$ and f are given as described.

Lemma 4.1. *B has minimal Turing degree.*

Proof. See [Ler83, V.2.6] $\square$

Suppose that g is a function which dominates f.

Lemma 4.2. $\langle T_k \rangle \leq_T (B \oplus g)'$.

Proof. Take $k = \mathbf{M}_\Phi$. Suppose we have already determined what T_k is. First, we wish to find out what kind of tree T_{k+1} is, which is the same as finding whether B Φ-splits on T_k or not.

We first ask $(B \oplus g)'$ if there is some n such that after running T_k for $g(n)$ many steps, no Φ-splits above $B \restriction n$ on T_k are found. If not, B splits on T_k. If there is such an n, then a search (recursive in $B \oplus g$) can find the least one n_0. We now ask $0'$ (which is below $(B \oplus g)'$) if there are any Φ-splits extending $B \restriction n_0$ on T_k. If not, then B does not split on T_k. If such a split exists, we repeat the process, asking if there is some $n_1 > n_0$ such that after $g(n_1)$ many steps no Φ-split extending $B \restriction n_1$ is found on T, and if one exists, we find it and ask $0'$, etc. This process must stop. For if B doesn't split on T_k then eventually we find some n such that there are no Φ-splits on T_k extending $B \restriction n$ (and of course none before $g(n)$.) If B does split on T_K, then the properties of f ensure that eventually, a Φ split extending $B \restriction n$ is always found before step $f(n)$, hence $g(n)$.

Now if B doesn't Φ-split on T_k then $T_{k+1} = T_k$. Otherwise, we need to find the least q such that $B \in [\mathrm{Sp}(T_k, \Phi, q)]$; We then know that $T_{k+1} = \mathrm{Sp}(T_k, \Phi, q)$.

Suppose that $T_k(q) \subset B$. Consider the following process: Let $T = \mathrm{Sp}(T_k, \Phi, q)$. $T(\langle\rangle) = T_k(q) \subset B$. Since every initial segment of B splits on T_k, both $T(0)$ and $T(1)$ are defined. We find them. If neither are initial segments of B, then the process halts. Otherwise, pick the one (say $T(0)$) such that $T(0) \subset B$. By the same reasoning, both $T(01)$ and $T(11)$ are defined, etc.

We see that this process is recursive in B. Thus B' can tell us whether it eventually terminates or not. However, a quick inspection shows that the process terminates iff $B \notin [T]$. Thus

$$\{r \,:\, B \in [\mathrm{Sp}(T_k, \Phi, q)]\}$$

is recursive in $B' \leq (B \oplus g)'$. We can thus inductively (on q such that $T_k(q) \subset B$) ask whether q is an element of the above set (this search is recursive in $B \oplus B' \equiv_T B'$). Eventually we will find the least one which is the desired q. $\square$

Now recall that we had a set A, falling under the third case, which excludes the second case. This means that for every function f recursive in D, there is some g recursive in A which dominates f. Thus we can go back and write A in each place we wrote g to get $\langle T_k \rangle \leq_T (A \oplus B)'$.

4.2 Construction I

The construction which yields the $B, \langle T_k \rangle$ and f with the desired properties is not difficult, and in fact differs only slightly from Jockusch's construction. We describe it now.

We construct a sequence β_s.

By the recursion theorem, we have a function $G(T, \Phi, s) \leq_T D$ such that for all (partial recursive function) trees T and Turing functionals Φ, $\lim_s G(T, \Phi, s)$ is the answer to the question

> Is there some t such that β_t is on T, and above β_t there are no Φ-splits on T?

At stage 0, we let $k(0) = 0$, $T_0[0] = \mathrm{id}$, $\beta_0 = \langle\rangle$ and $f(0) = 0$.

At stage $s+1$, we are given a sequence of trees $T_0[s], \ldots, T_{k(s)}[s]$ (each $T_{k+1}[s]$ is either equal to $T_k[s]$ or is a splitting subtree of $T_k[s]$), and β_s which is on every $T_k[s]$.

Using G we can define a function H, which tells us how far a requirement has to search until its guess (about whether B splits or not) has evidence in reality. Namely, if $k = \mathbf{M}_\Phi$, then $H(k, s)$ is defined to be the least stage $t > s$ such that either $G(T_k[s], \Phi, t) = \texttt{yes}$, or a Φ-split extending β_s is found on $T_k[s]$. By the properties of G, we know that such a t must exist. We also let $F(k, s) = \texttt{split}$ if a Φ-split above β_s is found on $T_k[s]$ at stage $H(k, s)$; otherwise we let $F(k, s) = \texttt{no split}$. Thus F gives us k's guess about whether B Φ-splits on $T_k[s]$. There is a *discrepancy* between reality and k's belief if $F(k, s) = \texttt{split}$ and $T_{k+1}[s] = T_k[s]$, or if $F(k, s) = \texttt{no split}$ but $T_{k+1}[s]$ is a splitting subtree of $T_k[s]$. If there is such a discrepancy, then it must be fixed by k by changing the tree T_{k+1} according to belief.

We thus say that k *requires attention* at s if there is a discrepancy between k's beliefs and reality. If there is some k requiring attention, we act as follows:

- If $F(k, s) = \texttt{no split}$, we let $T_{k+1}[s+1] = T_k[s]$.
- If $F(k, s) = \texttt{split}$, we want to define $T_{k+1}[s+1]$ to be $\mathrm{Sp}(T_k[s], \Phi, q)$ for some $q \subset p = T_k[s]^{-1}(\beta_s)$; as discussed above, we pick q to be the least such that β_s is *extended by some string* on $\mathrm{Sp}(T_k[s], \Phi, q)$.

  How do we effectively find this q? This is similar to the process in the proof of lemma 4.2: we inductively check $q = \langle\rangle, q = p \upharpoonright 1, q = p \upharpoonright 2, \ldots$. We know there is a split extending β_s; thus we can search for the least split on the tree $T_k[s]$ extending $T_k[s](q)$. If neither of the two strings of the split are compatible with β_s, this was the wrong q, and we try the next one. If one is, we repeat the process for that part of the split which is an initial segment of β_s. We halt with a "yes" answer (for the q checked) if we get to a string extending β_s.

In either case we let $k(s+1) = k+1$ and $T_l[s+1] = T_l[s]$ for $l \leq k$.

If no requirement asks for attention, we let $k(s+1) = k(s)+1$, $T_l[s+1] = T_l[s]$ for $l \leq k(s)$ and $T_{k(s)+1}[s+1] = T_{k(s)}[s]$.

Now either β_s is not on $T_{k(s+1)}[s+1]$, but properly extended by some string which is on this tree; or β_s is on $T_{k(s+1)}[s+1]$ and not terminal on it. Thus we can properly extend β_s to some β_{s+1} on that tree. That's the end of stage s.

We verify that the objects constructed indeed have the desired properties. We first show that the sequence of trees stabilizes.

Lemma 4.3. *For all k there is a stage $s^*(k)$ such that for all $t \geq s^*(k)$, $k(t) > k$.*

By the construction, we have that for all $t > s^*(k)$, $T_k[t] = T_k[s^*(k)]$; we let T_k be this final tree.

Proof. $s^*(0) = 1$. Assume $s^*(k)$ exists, we will show that $s^*(k+1)$ exists too. Suppose that $k = M_\Phi$. Suppose that after $t > s^*(k)$, $G(T_k, \Phi, s)$ has stabilized on the correct answer. Then we can let $s^*(k+1) = t$, because neither k, nor any stronger requirement, will seek attention after t. □

Corollary 4.4. *Each M_Φ succeeds.*

Proof. The guesses are eventually correct. Thus if B splits on T_k, then T_{k+1} is some splitting subtree of B; if B doesn't split on T_k then we're done anyway. □

Also as a corollary, we have that if B splits on T_k then eventually, at every s, a split above β_s on T_k is found at stage s of the construction; this allows us to calculate the function f as required.

Finally,

Lemma 4.5. *Assume that $k = M_\Phi$ and that B splits on T_k. Then $T_{k+1} = \mathrm{Sp}(T_k, \Phi, q)$ where q is the least string such that $B \in [\mathrm{Sp}(T_k, \Phi, q)]$.*

Proof. Say $T_{k+1} = \mathrm{Sp}(T_k, \Phi, q)$, and was finally defined at stage s. Of course $B \in [T_{k+1}]$. If $r \subsetneq q$ then at stage s we verified that β_s went off $\mathrm{Sp}(T_k, \Phi, r)$ (no string on that tree extends β_s), thus $B \notin [\mathrm{Sp}(T_k, \Phi, r)]$. □

4.3 Coding D

We now add more subtrees to our sequence T_k so that we will be able to recover D from $\langle T_k \rangle$.

We have new requirements $\mathbf{C}_n$, whose aim is to code the answer to $n \in D$? into the sequence of trees.

We organize all $\mathbf{C}_n$ and $\mathbf{M}_\Phi$ requirements effectively and identify each requirement with its place on the list.

For $\mathbf{C}_n$, we use two different kinds of narrow subtrees.

Definition. For $p \in 2^{<\omega}$, let $\mathrm{nar}_p, \mathrm{row}_p \colon 2^{<\omega} \to 2^{<\omega}$ be defined as follows

$$\begin{aligned} \mathrm{nar}_p(\sigma) &= p^\frown 00\sigma(0)00\sigma(1)00\sigma(2)\dots 00\sigma(|\sigma|-1) \\ \mathrm{row}_p(\sigma) &= p^\frown 11\sigma(0)11\sigma(1)11\sigma(2)\dots 11\sigma(|\sigma|-1). \end{aligned}$$

Now, suppose that T is a tree and that $p \in \operatorname{dom} T$. We let $\mathrm{Nar}(T,p) = T \circ \mathrm{nar}_p$ and $\mathrm{Row}(T,p) = T \circ \mathrm{row}_p$.

It is easy to see that if T is partial recursive, then so are $\mathrm{Nar}(T,p)$ and $\mathrm{Row}(T,p)$, and an index for each can be obtained uniformly from p and an index for T.

Lemma 4.6. *Suppose that T is a tree, and that $p, q \in \operatorname{dom} T$. Then*

$$[\mathrm{Nar}(T,p)] \cap [\mathrm{Row}(T,q)] = 0. \quad \square$$

This property motivates us to satisfy $k = \mathbf{C}_n$ by letting $T_{k+1} = \mathrm{Nar}(T_k, p)$ for some p if $n \in D$ and $T_{k+1} = \mathrm{Row}(T_k, p)$ for some p otherwise. As we did for the splitting subtrees, to be able to recover T_{k+1} from T_k and $(A \oplus B)'$ we eliminate the element of chance in the game by ensuring that $T_{k+1} = \mathrm{Nar}(T_k, p)$ for the least p such that $B \in [\mathrm{Nar}(T_k, p)]$ (or Row, as D decides).

Lemma 4.7. *Suppose that T is partial recursive and $X \in [T]$. Then*

$$\{p \in \operatorname{dom} T \,:\, X \in [\mathrm{Nar}(T,p)]\}$$

is recursive in X' (similarly for Row*).*

Proof. From X we can determine the $h \in 2^\omega$ such that $X \supset T[p^\frown h]$ (if $T(p) \not\subset X$ then X knows that p is not in the set). Then we simply ask if for such h and all $n \neq 0 \mod 3$, $h(n) = 0$ (1 for Row). $\square$

So indeed if we code as promised, then B' can find T_{k+1} from T_k by inductively asking, for longer and longer p, such that $T_k(p) \subset B$, whether $B \in [\mathrm{Nar}(T_k, p)]$ or $B \in \mathrm{Row}(T_k, p)$. The first "yes" answer we get is the correct one (and also tells us whether $n \in D$).

It would seem that all we need to do now is add this last element to the construction, namely if at stage s, $k = k(s) = \mathbf{C}_n$ and no stronger requirement asks for attention, then we should define $T_{k+1}[s]$ to be $\mathrm{Nar}(T_k[s], p)$ (or $\mathrm{Row}(T_k[s], p)$ if $n \notin D$) for the shortest p such that β_s is extended by some string on $\mathrm{Nar}(T_k[s], p)$. However, this is not so easy, for the reason that β_s may be nonterminal on $T_k[s]$ for some $k < k(s)$, but terminal on the narrow subtree $T_{k+1}[s]$; for $p = T_k^{-1}(\beta_s)$, $T_k(p^\frown 0)$ may be defined but $T_k[s](p^\frown 00)$ or $T_k[s](p^\frown 000)$ may be undefined. In this case we would not be able to extend β_s and would be stuck.

We note, however, that the reason that $T_k[s](p^\frown 00)$ or $T_k[s](p^\frown 000)$ are undefined, is that some split was not found on an earlier tree $T_l[s]$. The strategy to extricate ourselves from this situation is to try and extend β_s on $T_l[s]$ until we find the necessary split or guess that such a split does not exist. In the former case we can go on with the construction; in the latter we change our guess about T_{l+1} and remove later trees.

4.4 Construction II

We first describe the idea of the construction. The new construction attempts to follow in the footsteps of the previous construction; requirements $k = \mathbf{M}_\Phi$ define subtrees T_{k+1} which are splitting subtrees or not, and amend the trees according to their guesses. New requirements $k = \mathbf{C}_n$ will define T_{k+1} to be some narrow subtree of T_k, the type of which will be determined by whether $n \in D$ or not. All goes well as long as our initial segment $\beta = \beta_s$ is nonterminal on the trees. Suppose, however, that β is terminal on $T_{k[s]}[s]$, even after the action of $\mathbf{M}$ requirements. We then look at the greatest $l \leq k(s)$ such that $T_l[s]$ is a splitting subtree of $T_{l-1}[s]$. We note that β must be nonterminal on $T_l[s]$; otherwise, $l-1$ would have redefined T_l to be equal to T_{l-1}.

Let $p = T_l^{-1}\beta$. The reason that β is terminal on $T_{k(s)}$ is that for some q, $T_l(p^\frown q)$ is undefined; the latter is undefined because sufficiently many splits above β have not been discovered on T_{l-1}. If some extension γ of β on T_{l-1} really doesn't split on T_{l-1}, then we would like to extend β to γ and redefine $T_l = T_{l-1}$; this would resolve the issue. Otherwise, we will find more and more splits until β will be discovered to not be terminal on $T_{k(s)}$ after all. The question is how to tell whether there is such γ. After all, our guessing function G only answers questions about initial segments of B, not arbitrary nodes on the tree.

The solution is to gradually extend β, at substages of the stage s. As β is not terminal on T_l, we can extend it for one step on T_l. Now we search for splits on T_{l-1} as usual; if we believe there are none T_{l-1} can act and set $T_l = T_{l-1}$. Otherwise, a split is found which means we can extend β one step further on T_l. The process continues until β is nonterminal on $T_{k(s)}$, or a node γ as described is found and set as an initial segment of B.

We note a notational decision. It could be the case that at the beginning of the stage, some $k \in [l, k(s))$ redefines $T_{k+1} = T_k$ because β is terminal on T_k. Also, it could be the case that at some some substage we believe that β can no longer be extended on T_{l-1} (and redefine $T_l = T_{l-1}$); no previous tree is acting, which means that the problem which was just described has just reappeared, for a smaller pair $(l, k(s))$. The above actions will have to be repeated, but we do this at the next stage $s+1$ (we can move on because β is on all trees, even if terminal on some; between substages we will not change the trees, as the indexing would become cumbersome.)

We now give the full construction. At stage s we are given a sequence of trees $\langle T_k[s] \rangle_{k \leq k(s)}$ and a string β_t on all of those trees.

The guessing function $G(T, \Phi, s)$ is defined as before; it approximates the answer to the same question:

> Is there some t such that β_t is on T and there is no Φ-split extending β_t on T?

The elements of the construction are as follows:

Guessing Splitting

We define guessing functions H and F. At stage s, at a substage at which we have β_r as an initial segment of B, and for $k = \mathbf{M}_\Phi$, $H(k,s,r)$ is defined to be the least stage $u > r+s$ such that either $G(T_k[s],\Phi,u) = \texttt{yes}$, or a Φ-split extending β_r is found on $T_k[s]$ after running the tree for u many steps. $F(k,s,r)$ is defined analogously; it is $\texttt{split}$ in the former outcome and $\texttt{no split}$ in the latter.

Discrepancy between belief and reality is defined as in the previous construction; at the particular stage and substage, k feels the discrepancy if $F(k,s,r) = \texttt{split}$ but $T_{k+1}[s] = T_k[s]$, or if $F(k,s,r) = \texttt{no split}$ but $T_{k+1}[s+1]$ is a splitting subtree of $T_k[s]$.

At any stage and substage, if there is some requirement $k = \mathbf{M}_\Phi < k(s)$ asking for attention, then the strongest such requirement receives it and redefines T_{k+1} according to its belief F. As before, it lets $T_{k+1}[s+1] = T_k[s]$ if $F(k,s,t) = \texttt{no split}$, and $T_{k+1}[s+1] = \mathrm{Sp}(T_k[s],\Phi,p)$ for the shortest p such that β_r is extended by some string on the latter tree. It lets $k(s+1) = k+1$ and leaves the trees T_l for $l \leq k$ unchanged.

If β_r is not on $T_{k+1}[s+1]$ then it is extended by some string on that tree; we let β_{r+1} be such a string. Otherwise, we hand β_r to the next stage.

Action of the last tree

At some substage of stage s, we may instruct the last requirement $k(s)$ to act (this may happen if no other requirement asks for attention and if β_r is not terminal on $T_{k(s)}[s]$). $k(s)$ first extends β_r to some string β_{r+1} on $T_{k(s)}[s]$. Then,

- If $k(s) = \mathbf{M}_\Phi$, it lets $T_{k+1}[s+1] = T_k[s]$.
- If $k(s) = \mathbf{C}_n$ and $n \in D$, then $k(s)$ lets $T_{k+1}[s+1] = \mathrm{Nar}(T_k[s],p)$ for p the shortest substring of $T_k[s]^{-1}(\beta_{r+1})$ such that β_{r+1} is extended by some node on $\mathrm{Nar}(T_k[s],p)$. If necessary, β_{r+1} is extended again to β_{r+2} so that it actually lies on the new tree.
- If $k(s) = \mathbf{C}_n$ and $n \notin D$, we act as in the previous case, but with Row replacing Nar.

In both cases, we let $k(s+1) = k(s)+1$, and leave the trees T_l for $l \leq k(s)$ unchanged.

Construction

The instructions at stage s are as follows. If some $\mathbf{M}$ requirement asks for attention, we let the strongest one act and end the stage. If no $\mathbf{M}$ requirement asks for attention at the beginning of the stage, and if β_t is not terminal on $T_{k(s)}[s]$, then $k(s)$ is asked to act and to end the stage.

Otherwise, we let $T_l[s]$ be the last tree in our sequence which is a splitting subtree of its predecessor $T_{l-1}[s]$. There is a function h which is either the identity or a composition of functions of the form nar_q and row_q such that $T_{k(s)}[s] = T_l[s] \circ h$. Let $p = T_l^{-1}(\beta_t)$, $q = T_{k(s)}[s]^{-1}(\beta_t) = h^{-1}(p)$ and $r = h(q^\frown 0)$. Our aim is to extend β_t to be $T_l[s](r) = T_{k(s)}[s](q^\frown 0)$. This is done in substages.

As noted above, we must have β_t not terminal on $T_{l-1}[s]$, for we found a split on T_{l-1} extending β_t (or $l-1$ would have acted). Therefore also β_t is not terminal on $T_l[s]$. Thus we can extend it and let $\beta_{t+1} = T_l[s](r \upharpoonright |p|+1)$. We move to the first substage.

In general, at the beginning of substage i we have $\beta_{t+i} = T_l[s](r \upharpoonright |p|+i)$, already discovered to be on $T_l[s]$. If no $\mathbf{M}$ requirement wishes to act at this substage, then it must be that a split extending β_{t+i} was found on T_{l-1}; we can thus further extend this string on T_l and move to the next substage.

If at some substage an $\mathbf{M}$ requirement wishes to act, then that action ends the stage. Otherwise, at the end of the $i = |r|-|p|^{th}$ substage we have $\beta_{t+i} = T_l[s](r) = T_{k(s)}[s](p^\frown 0)$. (Note that during substages, β_{t+j} may be not a string on $T_{k(s)}[s]$, but at this last substage it is.) If β_{t+i} is terminal on $T_{k(s)}[s]$ we do not change any of the trees and move on to stage $s+1$ (we let $k(s+1) = k(s)$). Otherwise, we can let $k(s)$ act and end the stage.

4.5 Verifications II

The proof of lemma 4.3 goes through unaltered, noticing that if $k = \mathbf{C}_n$ then once T_k has stabilized, so will T_{k+1}, since k never acts to change the next tree.

As a corollary, we get that β_t gets extended infinitely often (so we do get a set B at the end). This is because if $k(s+1) > k(s)$ (i.e. if $k(s)$ gets to act), β_t has been extended at stage s (in fact, β_t has been extended at every stage except perhaps at stages at which some $\mathbf{M}$ requirement acts at the beginning of the stage, making $k(s+1) \leq k(s)$.)

We now may conclude that the construction succeeds; the rest of the verifications are as above: we showed that $\langle T_k \rangle \leq (A \oplus B)'$ and that D can be obtained from $\langle T_k \rangle$.

References

[Coo73] S. B. Cooper. Minimal degrees and the jump operator. *J. Symbolic Logic*, 38:249–271, 1973.

[Eps75] R. L. Epstein. Minimal degrees of unsolvability and the full approximation construction. *Mem. Amer. Math. Soc.*, 3(iss.1, 162):viii+136, 1975.

[Eps81] R. L. Epstein. Initial segments of degrees below $\mathbf{0}'$. *Mem. Amer. Math. Soc.*, 30(241):vi+102, 1981.

[Fri57] R. Friedberg. A criterion for completeness of degrees of unsolvability. *J. Symb. Logic*, 22:159–160, 1957.

[Ish99] S. Ishmukhametov. Weak recursive degrees and a problem of Spector. In *Recursion theory and complexity (Kazan, 1997)*, volume 2 of *de Gruyter Ser. Log. Appl.*, pages 81–87. de Gruyter, Berlin, 1999.

[Joc77] C. G. Jockusch, Jr. Simple proofs of some theorems on high degrees of unsolvability. *Canad. J. Math.*, 29(5):1072–1080, 1977.

[JP78] C. G. Jockusch, Jr. and D. B. Posner. Double jumps of minimal degrees. *J. Symbolic Logic*, 43(4):715–724, 1978.

[JS84] C. G. Jockusch, Jr. and R. A. Shore. Pseudojump operators. II. Transfinite iterations, hierarchies and minimal covers. *J. Symbolic Logic*, 49(4):1205–1236, 1984.

[KP54] S.C. Kleene and E.L. Post. The upper semi-lattice of the degrees of recursive unsolvability. *Annals of Mathematics*, 59:379–407, 1954.

[Ler83] M. Lerman. *Degrees of Unsolvability.* Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1983.

[Ler85] M. Lerman. On the ordering of classes in high/low hierarchies. In *Recursion theory week (Oberwolfach, 1984)*, volume 1141 of *Lecture Notes in Math.*, pages 260–270. Springer, Berlin, 1985.

[Ler86] M. Lerman. Degrees which do not bound minimal degrees. *Ann. Pure Appl. Logic*, 30(3):249–276, 1986.

[Lew03] A. E. M. Lewis. *Aspects of complementing in the Turing degrees.* PhD thesis, University of Leeds, 2003.

[Mar66] D. A. Martin. Classes of recursively enumerable sets and degrees of unsolvability. *Z. Math. Logik Grundlagen Math.*, 12:295–310, 1966.

[MM68] W. Miller and D. A. Martin. The degrees of hyperimmune sets. *Z. fur Math. Logik und Grund. der Math.*, 14:159–66, 1968.

[NSS98] A. Nies, R. A. Shore, and T. A. Slaman. Interpretability and definability in the recursively enumerable degrees. *Proc. London Math. Soc. (3)*, 77(2):241–291, 1998.

[Pos77] D. B. Posner. *High degrees.* PhD thesis, University of California, Berkeley, 1977.

[Pos81] D. B. Posner. The upper semilattice of degrees $\leq \mathbf{0}'$ is complemented. *J. Symbolic Logic*, 46(4):705–713, 1981.

[PR81] D. B. Posner and R. W. Robinson. Degrees joining to $\mathbf{0}'$. *J. Symbolic Logic*, 46(4):714–722, 1981.

[Rob72] R. W. Robinson. Degrees joining $\mathbf{0}'$. *Notices of the American Mathematical Society*, 19:A–615, 1972.

[Sac61] G. E. Sacks. A minimal degree less than $0'$. *Bull. Amer. Math. Soc.*, 67:416–419, 1961.

[Sas74] L. P. Sasso, Jr. A minimal degree not realizing least possible jump. *J. Symbolic Logic*, 39:571–574, 1974.

[SS89] T. A. Slaman and J. R. Steel. Complementation in the Turing degrees. *J. Symbolic Logic*, 54(1):160–176, 1989.

[SS99] R. A. Shore and T. A. Slaman. Defining the Turing jump. *Math. Res. Lett.*, 6(5-6):711–722, 1999.