Notre Dame Journal of Formal Logic
Volume 47, Number 3, 2006

The Hypersimple-Free C.E. WTT Degrees Are Dense in the C.E. WTT Degrees

George Barmpalias and Andrew E. M. Lewis

Abstract We show that in the c.e. weak truth table degrees if $\mathbf{b} < \mathbf{c}$ then there is an $\mathbf{a}$ which contains no hypersimple set and $\mathbf{b} < \mathbf{a} < \mathbf{c}$. We also show that for every $\mathbf{w} < \mathbf{c}$ in the c.e. wtt degrees such that $\mathbf{w}$ is hypersimple, there is a hypersimple $\mathbf{a}$ such that $\mathbf{w} < \mathbf{a} < \mathbf{c}$. On the other hand, we know that there are intervals which contain no hypersimple set.

1 Introduction

Hypersimple sets were introduced as a solution to Post's problem for the structure of the truth table degrees. Rogers observed that they were a natural solution to Post's problem for the weak truth table degrees as well. So it is interesting to know the distribution of these natural solutions in the weak truth table degrees. Moreover, weak truth table reducibility is the most appropriate for the study of hypersimplicity given that its essence is the existence of computable bounds (in the use of the relative computation) and hypersimplicity of a set A is based on the same notion: the *lack* of a computable sequence of bounds below which we get (strictly) more and more elements outside A. This connection becomes even clearer if we note that elements outside A below computable bounds are also important in a weak truth table reducibility since only elements not yet in A and below the use can rectify computations. We show that between any c.e. wtt degrees $\mathbf{b} < \mathbf{c}$ there is another c.e. wtt degree which does not contain any hypersimple set. We call such degrees *hypersimple-free*. Degrees which contain hypersimple sets are called *hypersimple*.

We also show that for every $\mathbf{w} < \mathbf{c}$ in the c.e. wtt degrees such that $\mathbf{w}$ is hypersimple, there is a hypersimple $\mathbf{a}$ such that $\mathbf{w} < \mathbf{a} < \mathbf{c}$. We know from Barmpalias [1] that there are nontrivial intervals $(\mathbf{b}, \mathbf{c})$ which contain no hypersimple sets. In fact, outside any nontrivial upper cone of Turing degrees we can find c.e. wtt degrees $\mathbf{b}$ such that no $\mathbf{c} \geq \mathbf{b}$ is hypersimple.

Received July 1, 2005; accepted February 1, 2006; printed November 14, 2006
2000 Mathematics Subject Classification: Primary, 03F60; Secondary, 03D30
Keywords: hypersimple, weak truth table degrees, density

In the following, all degrees will be c.e. and wtt. We use standard notation and when we describe a construction we assume a *current value* (corresponding to the current stage) for each of the various parameters involved. All sets will be c.e. and $A \leq_{\mathrm{wtt}} B$ is indicated as $\Phi^B = A; \varphi$ when we wish to make the algorithm (functional) Φ and the computable use φ of the reduction explicit. The use of computations made by wtt functionals is assumed to be strictly increasing. Finally, we use ℓ to denote the length of agreement of a potential reduction; for example, $\ell(\Phi^B = A; \varphi)$ is the length of agreement of $A \leq_{\mathrm{wtt}} B$ via the functional Φ and with use bounded by the partial computable function φ.

2 Hypersimple-free c.e. wtt Degrees

Theorem 2.1 *The hypersimple-free c.e. wtt degrees are dense in the c.e. wtt degrees. That is, if* $\mathbf{b} < \mathbf{c}$ *then there exists a c.e. hypersimple-free* $\mathbf{a}$ *such that* $\mathbf{b} < \mathbf{a} < \mathbf{c}$.

Proof By the density of the c.e. wtt degrees it is enough to show that for every $\mathbf{b} < \mathbf{c}$ there exists a hypersimple-free $\mathbf{a}$ with $\mathbf{b} \leq \mathbf{a} \leq \mathbf{c}$. Given the corresponding c.e. sets B, C we are going to construct a c.e. set A such that $B \leq_{\mathrm{wtt}} A \leq_{\mathrm{wtt}} C$ which is not equivalent to any hypersimple set. The first type of requirements guarantees that no hypersimple set is in $\mathbf{a}$:

$$\mathcal{Q}_{\Phi,\Psi,W} : \Phi^W = A; \varphi \text{ and } \Psi^A = W; \psi \Rightarrow \begin{cases} \exists (D_n)((D_n) \text{ is a sequence of} \\ \text{consecutive segments of } \mathbb{N} \wedge \\ \forall n(\overline{W} \cap D_n \neq \varnothing)) \end{cases}$$

where Φ, Ψ run over all partial computable functionals and W over all c.e. sets. Next we have the coding of B into A:

$$\mathcal{P} : B \leq_{\mathrm{wtt}} A.$$

This has highest priority and is satisfied in the usual way: we fix a coding (computable and 1-1) function $p : \mathbb{N} \to \mathbb{N}$ (e.g., $p(n) = 2n$) with coinfinite image (so that we have spare numbers to use for the other requirements). If every time that n appears in B we enumerate $p(n)$ into A then $\mathcal{P}$ is satisfied.

Finally we need to ensure that

$$\mathcal{N} : A \leq_{\mathrm{wtt}} C.$$

We will do this by equipping the construction with a C-permitting requirement regarding the enumerations into A. We begin by describing an atomic module for $\mathcal{Q}_{\Phi,\Psi,W}$. The idea behind this strategy is the following: assuming $\Phi^W = A; \varphi$ we enumerate a strong array (D_n) and try to achieve $\overline{W} \cap D_n \neq \varnothing$ for all n. The definition of each D_n is such that if all of its elements appear in W later on (giving $\overline{W} \cap D_n = \varnothing$) then by preserving a suitable segment of W via the additional hypothesis $\Psi^A = W; \psi$ we are able to ensure $\Phi^W \neq A; \varphi$ with a single diagonalization. There are two complications that must be taken into account here. First, if we consider all requirements, such a diagonalization must be C-permitted. Second, any diagonalization may later be canceled by a B-enumeration. But we can deal with these problems if we iterate the above procedure, use iterated diagonalizations for a single attack, and apply a C-permitting argument modulo B. Every D_n definition will be associated with a number of diagonalization witnesses $a_0, \ldots, a_n$ which will be used if and when $D_n \subseteq W$ occurs.

Let U be an infinite computable set especially for the use of a fixed $\mathcal{Q}$ strategy (i.e., disjoint from the set of B-codes V and the sets belonging to other $\mathcal{Q}$ strategies). Below, s is the current stage and any parameters mentioned in the construction are supposed to have a current value.

A_n (*n-attack setup*)

(1) Wait for $\ell(\Psi^A = W; \psi) > x$ for all $x \in \cup_{i<n} D_i$ and let u strictly bound the use of all these Ψ-computations.
(2) Pick a number of witnesses $a_0 < \cdots < a_t$ in $U - A$ such that $u < a_i$ where t is the number of B-codes (e.g., elements of $2\mathbb{N}$) in $\cup_{i<n} D_i$.
(3) Wait for $\ell(\Phi^W = A; \varphi) > a_t$ and define $D_n = \{\max \cup_{i<n} D_i + 1, \ldots, \varphi(a_t)\}$.

When A_n is run, D_i for $i < n$ are already defined. If $D_n \subseteq W$ later on, we will diagonalize by $a_i \searrow A$ and by imposing a finite restraint on A (in order to preserve a segment of W). This disagreement will be preserved unless a B-enumeration happens below u.

B_n (*D-failure step; in particular when the D-enumeration done in A_n has been proved wrong, i.e., $D_n \subseteq W$.*)

Consider the witnesses a_i and the use u which were defined in step A_n. Set $i = 0$.

(a) Wait until $\ell(\Psi^A = W; \psi) > x$ for all $x \in \cup_{j<n} D_j$ and $\ell(\Phi^W = A; \varphi) > a_i$.
(b) Restrain $A \upharpoonright u$ where u is the use of the Ψ-computations mentioned and suggest the enumeration $a_i \searrow A$. If in the meantime $A \upharpoonright u$ changes (e.g., by a B-enumeration) then B_n no longer suggests $a_i \searrow A$; upon next being passed control go to (a).
(c) If this enumeration is done by the construction later on, increase i by one and go to (a).

It will be $u < a_i$ as in step A_n. If we are permitted to put $a_i \searrow A$, this enumeration respects the A restraint we imposed in (a) above. This diagonalization can only be rectified via a W-enumeration below $\varphi(a_i)$. But no such enumeration can happen with elements in D_n because these are already in W (this made us start step B_n). And the elements in $\cup_{i<n} D_i$ are protected by an A-restraint. So if the strategy we are considering is not injured, the only reason why the disagreement is rectified is that $A \upharpoonright u$ changed; this may only happen due to a $B \upharpoonright u$ enumeration. According to the choice of t there will be at most t such B-enumerations—less than our witnesses. So either we succeed or some witness is not permitted.

The parts A_n, B_n above are only a piece of the whole $\mathcal{Q}$-strategy. We call them an AB-routine. A recursive iteration of AB-routines $AB(0)$, $AB(1)$, ... constitutes the $\mathcal{Q}$-strategy. We explain how a single AB-routine works. It enumerates its own array (D_n), which is a sequence of consecutive segments of (and potentially covering) $\mathbb{N}$. It starts by performing successively the steps $A_1, A_2, \ldots$. Each A_n defines D_n. It also finds suitable witnesses a_i for a "backup" diagonalization ripple planned in case $D_n \subseteq W$ later on, that is, in case the guess made in A_n (that D_n intersects the complement of W) is wrong.

After that, A_n has been completed and in order to pass to A_{n+1} we check whether $D_k \subseteq W$ holds for some $k \leq n$. In other words, whether or not one of the $\overline{W}$-guesses we have made so far (via (D_k)) looks incorrect. If not, then we can proceed to A_{n+1} in order to push (D_k) further. Otherwise for the least k with $D_k \subseteq W$ we pass control to B_k. No more steps apart from B_k will ever be performed in this AB-routine. B_k activates the backup diagonalization ripple with witnesses $a_0, \ldots, a_t$ prepared in A_k: it suggests (at some later suitable stage) a_0 as a witness for $\Phi^W \neq A$; φ and it also restrains (via Ψ and a restraint on A) the W-use of the computation (even after a possible $a_0 \searrow A$). If a_0 is C-permitted the disagreement may be rectified by a B enumeration, in which case we will pass to the next witness and so on. According to the choice of t we either succeed or one of our witnesses is not permitted. Note that in the atomic module above there are "wait" instructions. Taking into account that we may have to wait forever, the outcomes of an AB-routine are the following.

$\boxed{1}^{AB}$ As we go through $A_1, A_2, \ldots$ we get stuck in a wait instruction of some A_i and stay there forever. According to the wait conditions, this implies the satisfaction of $\mathcal{Q}$.

$\boxed{2}^{AB}$ Before passing to a next A_i we collapse onto a B_n-step. This does not automatically imply satisfaction of $\mathcal{Q}$ but it advances the implicit functional by which we wish to compute C from B in case some strategy fails to succeed (the permitting argument that will be used in the verification).

$\boxed{3}^{AB}$ We go through $A_1, A_2, \ldots$ with no permanent distraction. Under this outcome the AB-routine produces an infinite disjoint array (D_n) with $D_n \cap \overline{W} \neq \varnothing$ for all n, thus proving that W is not hypersimple (and $\mathcal{Q}$ is satisfied). There are no restraints associated with this outcome.

Iteration and coordination of AB-routines We say that an AB-routine requires attention if it is ready to perform the next step or it has stopped on a B_n step and the currently suggested witness a_i is permitted (i.e., $C \upharpoonright a_i$ has changed this stage). Turning to the whole $\mathcal{Q}$-strategy, we start by executing $AB(0)$ (which is identical to the typical AB-routine described above) and continue as follows in an inductive mode. Every time we access $\mathcal{Q}$ we consider the largest t such that we have previously run $AB(t)$. If some $AB(i)$, $i \leq t$ requires attention, we pass control to the least one; if it required attention because its suggested witness is permitted, we enumerate that witness into A. If no $AB(i)$, $i \leq t$ requires attention, we access $AB(t)$; if it is already in the stage of a suggested witness under a B_n step we start $AB(t+1)$ with the additional restriction that all the a_n-witnesses chosen during its A_n-steps are larger than the witnesses already suggested by the $AB(i)$ for $i \leq t$, when they terminated, and larger than the restraints imposed by these routines.

Injury among AB-routines Note that when an AB routine performs an A-enumeration it may interfere with the restraints of lower AB-routines which have reached a B_t step. In this case we initialize those steps as follows: we go to part (a) and set the parameter i equal to the least index such that $a_i \notin A$ yet. This does not affect our argument: if a diagonalization of AB is rectified because of a diagonalization of a higher AB', before it enumerates the next witness there must be a $B \upharpoonright u$ change which will rectify the higher diagonalization.

***C*-permitting and outcomes** From the above, any enumeration into A is C-permitted and so $A \leq_{\text{wtt}} C$. Note that as we go through $AB(1), AB(2), \ldots$, we build on more and more restraints on A. Since a diagonalization can only be rectified by a B-change below computable bounds, if C is indeed noncomputable from B some diagonalization will succeed. So the production of AB routines will stop (since they require Φ-agreement on larger and larger segments) and the restraints will end up finite. The outcomes of the entire $\mathcal{Q}$-strategy are the following.

$\boxed{1}^{\mathcal{Q}}$ As we go through $AB(1), AB(2), \ldots$ we get stuck in a wait instruction of some $AB(i)$ and stay there forever. Or some $AB(i)$ never stops running. Either case implies the satisfaction of $\mathcal{Q}$ as before and also that the overall A-restraints that $\mathcal{Q}$ imposes are bounded (i.e., finite).

$\boxed{2}^{\mathcal{Q}}$ We never stop running $AB(1), AB(2), \ldots$. By permitting (see below) this means that we can compute C from B: there are infinitely many witnesses suggested and if some n enters C it will permit a diagonalization. This must be rectified and so a B enumeration must happen below computable bounds.

These outcomes show that our strategy is successful. Moreover, it is not difficult to see that all $\mathcal{Q}$ strategies can work together with only a finite injury effect. Whenever some $\mathcal{Q}$ acts (i.e., enumerates a number into A or increases its restraints) it initializes all lower requirements. But according to the outcomes above it acts only finitely often (imposing a final finite A-restraint) and so it allows lower priority requirements (which respect the higher priority A-restraint) to be satisfied. When a $\mathcal{Q}$ strategy is initialized, it chooses witnesses larger than any witness chosen so far by any $\mathcal{Q}$ strategy in the construction. Our priority list is

$$\mathcal{P} > \mathcal{Q}_0 > \mathcal{Q}_1 > \cdots$$

where $(\mathcal{Q}_i)$ is an effective list of the $\mathcal{Q}$ requirements.

Construction At stage s, first check whether any B-enumeration happened. If $n \searrow B$ at s, put $p(n) \searrow A$. Then start accessing $\mathcal{Q}_i$, $i = 0, \ldots$ successively until one of them (say $\mathcal{Q}_t$) acts (i.e., enumerates a number into A or increases its restraints) or we reach $\mathcal{Q}_s$. Initialize $\mathcal{Q}_j$ for $j > t$.

Verification The following lemmas show that the construction produces a set A which satisfies the requirements.

Lemma 2.2 $A \leq_{wtt} C$.

Proof Suppose that $\Phi^C = B; \varphi$. We want to answer "$n \in A$?" using $C \upharpoonright \varphi(n)$. If $n \in p(\mathbb{N})$ we can effectively answer by asking $B \upharpoonright n$ (and so, by asking $C \upharpoonright \varphi(n)$). If $n \in U$, that is, the special set used by some $\mathcal{Q}$ requirement, we only need to choose a stage s after $C \upharpoonright n$ has taken its final value and we can be sure that $n \notin A$ unless it is already there by that stage. Finally, if n does not belong to any U either, we can conclude that $n \notin A$. □

Lemma 2.3 $B \leq_{wtt} A$.

Proof Since $\mathcal{P}$ has highest priority, $n \in B \Leftrightarrow p(n) \in A$. □

Lemma 2.4 *Suppose that $\mathcal{Q}$ stops being injured by higher priority requirements. Then for each of its AB-routines which collapses onto a B-step the following holds: if it has enumerated a witness into A, it will not enumerate a new witness unless there is a change in $B \upharpoonright u$ (where u is the parameter of that particular B-step of the routine). So the available witnesses of AB (including the current one) will always be more than the elements of $\mathbb{N} - B$ below u.*

Proof For the first diagonalization it holds trivially. If it holds up to some diagonalization we argue as follows: an extra witness will not be used unless the current disagreement is spoiled. This can only happen by a $B \upharpoonright u$ change or an A-enumeration by a higher AB'-routine. In the first case the claim holds. In the second case a disagreement will be created on the same reduction. The next witness of AB cannot be used unless that higher disagreement is rectified (because we require an expansionary stage). Now the higher disagreement can only be rectified by a $B \upharpoonright u$-change or by an even higher diagonalization and so on. Since there are only finitely many AB'-routines higher than AB this procedure must resolve with a $B \upharpoonright u$-change. This concludes the argument. □

Lemma 2.5 *Each $\mathcal{Q}$ requirement is satisfied and acts only finitely many times.*

Proof In an inductive fashion, assume we are in a stage where all higher priority requirements than $\mathcal{Q}$ (apart from $\mathcal{P}$) have stopped requiring attention and are satisfied. If $\mathcal{Q}$ is not satisfied, its strategy will run forever, producing infinitely many AB routines; each of them will terminate on a B_n step, thus passing control to a multiple diagonalization system which suggests A-enumerations (otherwise W is not hypersimple and so it is satisfied). Some of these witnesses are used, but their diagonalizations may be rectified through B-enumerations. The crucial point here is that each multiple diagonalization system (i.e., the B_t-step in which each AB-routine stopped) has to stop because one of its witnesses does not get C-permission, provided that the functionals Φ, Ψ of the requirement are total.

By a permitting argument we are going to show that if none of the diagonalizations are permanent (i.e., the requirement is not satisfied), then we are able to compute C from B (in a wtt way). In order to define the computable use of the reduction we associate each $n \in \mathbb{N}$ with the least AB routine (in particular, its multiple diagonalization system B_t) which has all of its witnesses $> n$. The use of the reduction on n will be the parameter u of step B_t. Note that a diagonalization from this system can only be rectified either by a $B \upharpoonright u$ change or by a higher diagonalization (by some higher AB-routine) whose rectification will also require a $B \upharpoonright u$ change. To compute "$n \in C$?" we wait until a stage s_0 where step B_t suggests a witness (this will happen since $\mathcal{Q}$ is not satisfied and because of Lemma 2.4). Then we enumerate the axiom

$$n \in C \iff n \in C[s_0]$$

with use $B \upharpoonright u$. If $n \searrow C$ at a later stage s_1 consider the following cases.

Case (a): Some diagonalization happens by higher AB-routines in the interval $(s_0, s_1]$. This disagreement can only be rectified by a $B \upharpoonright u$ change or by the creation of a higher disagreement (i.e., at a smaller initial segment) by a higher AB. But then the higher disagreement can only be rectified by a $B \upharpoonright u$ change or by the creation of an even higher disagreement and so on. Since there are only finite number of AB

routines preceding the one that n is associated with, this must resolve to a $B \upharpoonright u$ change.

Case (b): If no action is performed by higher AB-routines and $B \upharpoonright u$ is intact, the current witness of B_t will still be valid and it will be permitted at s_1. Now *this* disagreement must be rectified and so a $B \upharpoonright u$-change must occur later on.

In any case there will be a change in $B \upharpoonright u$ and this shows that $C \leq_{\text{wtt}} B$, a contradiction. □

This completes the proof.

3 Hypersimple Degrees

Theorem 3.1 *If* $\mathbf{w} < \mathbf{c}$ *in the c.e. wtt degrees and* $\mathbf{w}$ *is hypersimple, then there is a hypersimple* $\mathbf{a}$ *such that* $\mathbf{w} < \mathbf{a} < \mathbf{c}$.

Proof We have seen in Barmpalias [1] that there is a certain type of conflict when we try to construct a hypersimple set A above a given W, and sometimes this makes such a construction impossible. In particular, there are a lot of degrees (outside any nontrivial Turing upper cone) which are not bounded by any hypersimple degree. Below we will see how to manage this conflict when we have the information that W is hypersimple. If D_n is an effective enumeration of all finite sets and (Φ, φ) runs over an effective enumeration of all partial computable functionals/functions then the following requirements guarantee the result:

$$\begin{aligned} \mathcal{N}: &\quad A \leq_{\text{wtt}} C \\ \mathcal{P}: &\quad W \leq_{\text{m}} A \\ \mathcal{S}_{\Phi,\varphi}: &\quad \Phi^W \neq A; \varphi \\ \mathcal{R}_\varphi: &\quad \exists n(D_{\varphi(n)} \subseteq A) \ \vee \ D_\varphi \text{ is not a strong array.} \end{aligned}$$

We say that D_φ is a strong array if φ is computable and for $n \neq m$, $D_{\varphi(n)} \cap D_{\varphi(m)} = \varnothing$. Notice that $\mathcal{P}$ asks for something stronger than we really need, namely, m-reducibility instead of wtt. Fix a computable $c : \mathbb{N} \mapsto \mathbb{N}$ which is 1-1 and such that $\mathbb{N} - c(\mathbb{N})$ is infinite (e.g., $c(n) = 2n$). We will arrange that

$$n \in W \iff c(n) \in A,$$

thus satisfying $\mathcal{P}$. Assume a priority list where $\mathcal{P}$ has highest priority and the infinitely many $\mathcal{S}_{\Phi,\varphi}$, $\mathcal{R}_\varphi$ follow in an effective way (based on the effective enumeration of (Φ, φ) that we assumed earlier). Each of $\mathcal{S}_{\Phi,\varphi}$, $\mathcal{R}_\varphi$ will be finitary (i.e., act finitely often) and any A-enumeration they do must not injure $\mathcal{P}$ in any way. An A-enumeration affects $\mathcal{P}$ only when it involves c-codes, that is, elements in $c(\mathbb{N})$.

$\mathcal{S}_{\Phi,\varphi}$ strategy As usual, we can assume that φ is strictly monotone. We will use Sacks coding in order to code enough information from C so that W cannot wtt-compute A (given that W cannot wtt-compute C). The problem is that the codes may be used by hypersimplicity requirements. So we must restrain them and every time this restraint is violated we have to start the strategy from the beginning. Since the Sacks coding will leave the length of agreement bounded (Φ being a wtt functional) our restraints will be finite.

The strategy works as follows: first, if some $n \searrow C$ in the current stage and n is associated with an active code, it enumerates the code into A. Second, it finds

the least $i \notin C$ and $i < \ell(\Phi^W = A; \varphi)$ for which it has not assigned a code and it assigns the least number which is

(1) larger than the restraints of higher priority requirements,
(2) not in A,
(3) not in $c(\mathbb{N})$,
(4) larger than i.

This code will be *active* when it is below $\ell(\Phi^W = A; \varphi)$ and *inactive* otherwise. It restrains all the codes it has defined so far and it requires attention when some $n \searrow C$ with an active code or at expansionary stages (when $\ell(\Phi^W = A; \varphi)$ is larger than ever before). Notice that once a code has been assigned it is permanent (although it may be inactive).

$\mathcal{R}_\varphi$ strategy Although it was easy to find a strategy for $\mathcal{S}$ which does not interfere seriously with $\mathcal{P}$, it is more difficult to do the same with $\mathcal{R}$, since hypersimplicity requirements cannot afford to choose their witnesses from a prearranged computable set. So we have to allow them to enumerate into elements of $c(\mathbb{N})$ as well and to avoid the destruction of $\mathcal{P}$ we will take advantage of the hypersimplicity of W. Based on the given strong array $(D_{\varphi(n)})$ (which tries to show that A is not hypersimple) we will construct a strong array (G_n) which tries to show that W is not hypersimple. When (G_n) fails, that is, $G_k \subseteq W$ for some k, we will cause a $(D_{\varphi(n)})$-failure (i.e., $D_{\varphi(k)} \subseteq A$ for some k) without creating any potential problems to $\mathcal{P}$. Note that (G_n) will definitely fail since W is given hypersimple. To be more specific, we simply define

$$G_n := \{k \mid c(k) \in D_{\varphi(n)}\}.$$

Now since W is hypersimple, some $G_n \subseteq W$ at some stage (in fact this will happen for infinitely many n). But then $\mathcal{R}_\varphi$ can be satisfied by putting into A only the elements in $D_{\varphi(n)} - c(\mathbb{N})$; indeed, $c(\mathbb{N}) \cap D_{\varphi(n)}$ is already in A by $\mathcal{P}$'s module and $G_n \subseteq W$. In other words we satisfy $\mathcal{R}$ without enumerating into A any c-codes (such an enumeration is left to $\mathcal{P}$). Of course, since we want to build $A \leq_{\mathrm{wtt}} C$ such a $D_{\varphi(n)}$ enumeration must be C-permitted. So the $\mathcal{R}$ strategy works by suggesting various $D_{\varphi(n)}$, $n \in \mathbb{N}$ for A-enumeration (those with $G_n \subseteq W$) and it enumerates the first one which is C-permitted (i.e., there is a $C \upharpoonright \min D_{\varphi(n)}$ change on the current stage) and all of its elements are greater than the current restraint. It requires attention every time it has a new suggestion D to make or an enumeration to perform of an existing suggestion D.

Construction In order to let all the strategies work together we only need to make sure that lower priority requirements respect the restraint r set by higher ones. The only strategies which impose restraints are $\mathcal{S}$ which restrain their codes. Whenever an $\mathcal{S}$ or $\mathcal{R}$ receives attention we initialize all lower priority requirements. Every $\mathcal{S}$ chooses codes greater than the restraint r; $\mathcal{R}$ only enumerates into A a $D_{\varphi(n)}$ with all members greater than r. The *construction* is, at stage s,

1. for every n, if $n \in W$ (and $c(n) \notin A$) put $c(n) \searrow A$;
2. find the least $\mathcal{S}$ or $\mathcal{R}$ which requires attention and run the relevant strategy; initialize the lower priority requirements.

Verification

Lemma 3.2 $A \leq_{wtt} C$.

Proof To answer "$n \in A$?" we first check whether $n \in c(\mathbb{N})$. If yes, we can easily C-answer it since $W <_{\text{wtt}} C$ (no strategy, apart from $\mathcal{P}$, enumerates c-codes into A). If not, we wait until a stage s_0 larger than n where $C \upharpoonright n$ is correct. If n is not in A by that stage, it is not going to be in later. Indeed, by the C-permitting we require in the construction, no $\mathcal{R}$ requirement will enumerate n into A after s_0. Also, n is not going to be enumerated into C as a code of some $\mathcal{S}$ since this can only happen at a stage where $C \upharpoonright n$ changes. □

Lemma 3.3 $W \leq_{wtt} A$.

Proof According to the c-coding we only need to show that if $n \notin W$ at some stage then $c(n) \notin A$ at the same stage. This holds as the $\mathcal{R}$ requirements (and of course the $\mathcal{S}$ requirements) do not enumerate any c-codes in A (unless they are already in A). □

Lemma 3.4 *All $\mathcal{S}$, $\mathcal{R}$ requirements are satisfied and stop requiring attention after a certain stage.*

Proof Assume that all higher strategies have stopped requiring attention. If $\mathcal{S}$ was not satisfied we show how to wtt-compute C from W: to answer "$n \in C$?" wait until the strategy associates n with a code or $n \searrow C$. This will happen since $\ell(\Phi^W = A; \varphi) \to \infty$. If t is the code then wait until a stage where $W \upharpoonright \varphi(t)$ is true and $\ell(\Phi^W = A; \varphi) > t$. From now on t will be active and so n cannot enter C later on since t would create a permanent disagreement on $\Phi^W = A$, a contradiction. Since $C \not\leq_{\text{wtt}} W$, $\ell(\Phi^W = A; \varphi)$ must come to a limit and so $\mathcal{S}$ is satisfied and stops requiring attention.

In the case of $\mathcal{R}$ we know that infinitely many D will be suggested for enumeration into W. If C failed to give the permitting required in order to satisfy $\mathcal{R}$ it would be computable. □

This concludes the proof of the theorem.

References

[1] Barmpalias, G., "Hypersimplicity and semicomputability in the weak truth table degrees," *Archive for Mathematical Logic*, vol. 44 (2005), pp. 1045–65. Zbl 1077.03026. MR 2193189. 361, 367

Acknowledgments

Barmpalias was supported by EPSRC Research Grant No. EP/C001389/1. Lewis was supported by EPSRC grant No. GR /S28730/01. Both authors were partially supported by the NSFC Grand International Joint Project, No. 60310213, "New Directions in the Theory and Applications of Models of Computation." The authors would like to thank Wei Wang and Decheng Ding for their hospitality in the Nanjing University in May 2005 where this work started; we also thank Angsheng Li who supported our visit to China.

School of Mathematics
University of Leeds
Leeds LS2 9JT
UNITED KINGDOM
georgeb@maths.leeds.ac.uk
thelewisboy@hotmail.com