Notre Dame Journal of Formal Logic
Volume 45, Number 3, 2004

There Are No Maximal Low D.C.E. Degrees

Rod Downey and Liang Yu

Abstract We prove that there is no maximal low d.c.e degree.

1 Introduction

A natural extension of the notion of a computably enumerable (c.e.) set is that of a d.c.e. set which is a set obtained as the difference of two c.e. sets $A = W - V$. Equivalently, a d.c.e set A is a set for which there exists a computable function $f(x, s)$ so that $A(x) = \lim_s f(x, s)$, $f(x, 0) = 0$, and $\forall x |\{s, f(x, s) \neq f(x, s+1)\}| \leq 2$. As well as being interesting in their own right, the d.c.e. Turing degrees can be studied both to give insight into the c.e. Turing degrees and into the Δ^0_2 degrees. The investigation of the present paper can be viewed as contributing to all three of these goals.

The uppersemilattice of the d.c.e. degrees is not elementarily equivalent to that of the c.e. degrees by Arslanov [1] and Downey [5]. Perhaps the most striking difference between the d.c.e. degrees and the c.e. degrees comes from the following two theorems.

Theorem 1.1 (Sacks [9]) *The c.e. degrees are dense.*

Theorem 1.2 (Cooper, Harrington, Lachlan, Lempp, Soare [4]) *The d.c.e. degrees are not dense. Indeed, there is a* maximal *d.c.e. degree* $\mathbf{a}$. *That is,* $\mathbf{a} < \mathbf{0}'$, *and there are no d.c.e. degrees* $\mathbf{b}$ *with* $\mathbf{a} < \mathbf{b} < \mathbf{0}'$.

Also notice that density properties allow us to compare the d.c.e. degrees and the Δ^0_2 degrees. By an unpublished result of Lachlan, there are no minimal d.c.e. degrees, yet Sacks constructed a minimal Δ^0_2 degree. Actually there is a very interesting theme here that "toward $\mathbf{0}$" the d.c.e. degrees are like the c.e. degrees and "toward $\mathbf{0}'$" they resemble more the Δ^0_2 degrees.[1]

Received February 7, 2004; accepted June 16, 2004; printed October 26, 2004
2000 Mathematics Subject Classification: Primary, 03D28
Keywords: D.c.e. degrees, Turing degrees

One of the fundamental operators in computability theory is the *jump* operator. Quite early on it was found that there were noncomputable sets which were indistinguishable from the computable sets by the jump operator.

Definition 1.3 A set A is low if $A' \equiv_T \varnothing'$.

A recurrent theme in computability theory, and particularly the study of the c.e. sets and degrees, is that low sets should resemble computable sets in their properties. Technically, many results in this vein rely in one form or another on a method invented by Robinson. Robinson proved the following theorem, which is a combination of the well-known Sacks Splitting Theorem and Density Theorem.

Theorem 1.4 (Robinson [8]) *For any low c.e. set L and c.e. set $A >_T L$, there are two c.e. sets B_0, B_1 so that $L <_T B_0, B_1 <_T A$ and $A \equiv_T B_0 \oplus B_1$.*

The lowness hypothesis of L in Robinson's Theorem is necessary as witnessed by Lachlan's Nonsplitting Theorem [7].

Robinson's Theorem introduced the technique, now called the *Robinson technique*, which allows us to use lowness for c.e. sets. We will discuss this technique in detail in the proof of our main result. Here it suffices to say that the technique used the lowness of L to, in the limit, answer Σ_1^L questions within the construction, and *relied on the enumerability of L to "certify" certain "no" answers within the construction.* (This will be explained in detail in the construction below.) Recently, Arslanov, Cooper, and Li [2] claimed a sweeping generalization of the Robinson technique by claiming that Theorem 1.4 could be proven *without* the hypothesis that L is c.e. Unfortunately their proof contains a fatal flaw.

We do not know if their claimed result in its full generality is correct.

Question 1.5 *Given a low set A and a c.e. set B with $A \leq_T B$, do there exist c.e. sets B_1, B_2 with $B_1 \oplus B_2 \equiv_T B$ and $B_1 \oplus A|_T B_2 \oplus A$? What about $B = \varnothing'$?*

One of the consequences claimed by Arslanov, Cooper, and Li was the following.

Theorem 1.6 *There is no maximal low d.c.e. degree.*

It is the goal of the present paper to give a proof of Theorem 1.6. We believe that our proof of Theorem 1.6 is interesting in its own right as it introduces a method of applying the Robinson technique outside of the c.e. degrees and relies upon special properties of the d.c.e. degrees to allow its application. Our methods do not seem to allow us to split $\mathbf{0}'$ over all low d.c.e. degrees, and hence the following question weakening Question 1.5 also suggested by the Arslanov-Cooper-Li claims remains.[2]

Question 1.7 *For any low d.c.e. set L, is there a c.e. splitting $A_0 \oplus A_1 = \varnothing'$ so that $A_i \oplus L <_T \varnothing'$?*

2 Intuition of the Proof of Theorem 1.6

2.1 The Robinson technique for c.e. sets We remind the reader how the Robinson technique works for the Sacks Splitting Theorem and with L c.e. We need a lemma.

Lemma 2.1 *For any low set L, $X(L) \leq_T \varnothing'$ where $X(L) = \{j : \exists n \in W_j \exists m(n = L \upharpoonright m)\}$.*

The classical application of the Robinson technique is to split any c.e. set A over a low c.e. set L, meeting Sacks type requirements of the form

$$N_e^i : A_i \neq \Phi_e^{A_{1-i} \oplus L}.$$

The basic idea is the same as for the Sacks Splitting Theorem. At a stage s, if we see $\ell(e, i, s) > x$ where $\ell(e, i, s) = \max\{y : \forall z < y : \Phi_e^{A_{1-i} \oplus L}(z) = A_i(z)[s]\}$, we will attempt to preserve $A_{1-i} \oplus L \upharpoonright \varphi(x)[s]$. We do this by asking that elements below this use entering A after stage s should be directed into A_i and not A_{1-i}. Then we so preserve the use of the left-hand side and argue that, if we fail to diagonalize, then A is computable, since eventually all but a computable part of A will be directed into A_i rather than A_{1-i}. However, there is a slight problem with this plan. The set L is not under *our* control. *We* can preserve $A_{1-i}[s] \upharpoonright \varphi_e(\ell(e, i, s))[s]$ as much as we like, but it is up to the opponent to decide whether this *also* preserves $A_{1-i} \oplus L \upharpoonright \varphi_e(\ell(e, i, s))[s]$. The problem is that *if* we preserve this computation, then the use might be L-incorrect. However, we will have directed some small numbers perhaps into A_i, which might fatally injure some lower priority requirement trying to preserve A_i. This key insight can be turned around into a proof that not every c.e. degree can be split over all lesser ones [7].

Here is where we use the fact that L is low *and c.e.* Since L is low, by the Limit Lemma, there is a computable function $g(j, s)$ so that for every j, $\lim_s g(j, s) = X(L)(j)$. For each argument x, Robinson's idea is to build a computably enumerable set $U_x = W_{j(x)}$ whose index is given by the Recursion Theorem. This set allows us to use $X(L)$ to "L-certify" computations as follows. Suppose, as above we see $\ell(e, i, s) > x$. We need to decide if we should preserve the left-hand side of the computation. Our action would be to put the index n of $L \upharpoonright \varphi_e(x)[s]$ into the test set U_x. By waiting or speeding up the L-enumeration we may assume that n immediately enters U_x. Now if this L-configuration is correct, $g(j(x), s)$ should eventually output 1. Thus we can now mark time and run the enumerations of g and L until a stage $t \geq s$ is found when *either* $L_t \upharpoonright \varphi_e(x)[s] \neq L_s \upharpoonright \varphi_e(x)[s]$, *or* $g(j(x), s) = 1$. In the latter case, we will declare the computation to be L-certified and impose restraint. In the former case, we see that the computation $\Phi_e^{A_{1-i} \oplus L}(x)[s]$ now appears wrong. *Furthermore, since L is assumed to be c.e., we actually know that for all $t' \geq t$, $L_{t'} \upharpoonright (\varphi_e(x)[s]) \neq L_s \upharpoonright (\varphi_e(x)[s])$.* In this case we impose no restraint for any $y \geq x$. We would repeat this process each time we see $\ell(e, i, s) > x$. If we see infinitely many L-certified computations we actually restrain, then we will impose only finitely much overall restraint for x, and we can show that the overall restraint is finite for a fixed N_e^i.

2.2 The problem where L is Δ_2^0. We can still try to use a process as above with L no longer c.e. Indeed we can have an enumeration of L given by the Limit Lemma, $L = \lim_s L_s$, meaning that for each z $L_s(z) \neq L_{s+1}(z)$ only finitely often.

Imagine that we attempt the above construction with L simply low but not c.e. At some stage we again see $\ell(e, i, s) > x$. Again we need to decide whether to impose restraint on $A_{1-i} \upharpoonright \varphi_e(x)[s]$. Of course, we can put n as above into some test set U_x. If we receive a "yes" answer with an L-certified computation then, as before, we could impose restraint. But suppose that we get a "no" answer at t. That means that $L_t \upharpoonright (\varphi_e(x)[s]) \neq L_s \upharpoonright (\varphi_e(x)[s])$. We have a choice. Should we impose restraint or not?

If we do impose restraint, then we are back to square one. Now the restraint could be infinite since $\Phi_e^{A_{1-i}\oplus L}(x) \neq A_i(x)$ because of unbounded use on the left-hand side. Thus the overall restraint could restrain the noncomputable part of A from A_{1-i}, killing lower priority requirements.

If we don't impose restraint (the method suggested by Arslanov, Cooper, and Li), then perhaps small numbers enter A_i at stage t. However, perhaps *really* $L \upharpoonright (\varphi_e(x)[s]) = L_s \upharpoonright (\varphi_e(x)[s])$. This is because, quite distinct from the c.e. case, $L_{t'} \upharpoonright (\varphi_e(x)[s]) = L_s \upharpoonright (\varphi_e(x)[s])$ for some $t' > t$. Perhaps elements enter and then leave, or vice versa. But now, since we did not impose restraint at stage t, A_{1-i} might have changed. The crucial point is that now the set U_x is *useless*. That is, now U_x really does contain an index of a prefix (an initial segment of) L, and henceforth $g(j(x), s)$ can simply return 1.

There does seem to be a class of low sets for which the Robinson technique seems to work. These low sets are sets with not only a lowness certification, but a "low enumeration." We will explore this idea elsewhere.

2.3 The proof of our theorem We will assume that we are given a low d.c.e. set L. We will construct a Δ_2^0 set Δ and a d.c.e. set A so that $\Delta \not\leq_T L \oplus A$ and $A \not\leq_T L$. We satisfy the requirements below.

$$M_e \ : \ A \neq \Psi_e^L.$$

$$N_e \ : \ \Delta \neq \Phi_e^{L\oplus A}.$$

The easiest requirements to deal with are the M-type requirements. For the M-type requirements, we apply the Robinson technique and Friedberg-Muchnik strategy. The only action for such requirements is to put some numbers into A and, in the full construction, restrain those numbers from leaving A.[3]

The action of M_e is the following. We pick a follower x and wait until $\Psi_e^L(x) = 0$ at some stage s. As above, we put the L-use $n = L_s \upharpoonright \psi_e(x)$ into U_e. Precisely as above, we find the least $t \geq s$ such that either $n \neq L_t \upharpoonright \psi_{e,s}(x)$, or $g(j, t) = 1$, in which case we L-*certify* the computation and put x into A_t. We will protect this number's removal from A with priority e. We put x into A_t only when the computation is L-certified. It may later happen that we were wrong, but this happens at most finitely often by the definition of g. Notice that once we put x into A, we only need to pick a new follower x' if we see some stage s' with $\Psi_e^L(z) = A(z)[s']$ for all $z \leq x$. That is, inductively *all* the followers we have put into A for the sake of M_e must be incorrect. This entails that *all* of the apparent initial segments of L ever put into U_e must also be wrong. This happens at most finitely many times. The point is there is at most *one* $n \in U_{x,s}$ so that $n = L_s \upharpoonright m$ for some m at any stage s. The usual argument for this is to ensure that U is a prefix free set. There are some minor problems to ensure this.

The argument for the N-type requirements is significantly more subtle. For N-type requirements, we also apply the Robinson technique and the Friedberg-Muchnik strategy. We never put any numbers into (but may extract some numbers from) A for these requirements. For every requirement N_e, we try to ensure that $\Delta(x) \neq \Phi_e^{L\oplus A(x)}$ for some x by putting x into (or pulling x out of) Δ at most finitely often. Fixing x, we try to ensure $\Delta(x) \neq \Phi_e^{A\oplus L}(x)$. Let $n = L_s \upharpoonright \varphi_{e,s}(x)$. Again we enumerate n into a c.e. set V which we shall build during the construction. Again we may assume that we have in advance an index j such that $V = W_j$. Again we find the least $t \geq s$ such that either $n \neq L_t \upharpoonright \varphi_{e,s}(x)$, or $g(j,t) = 1$. In the case that $g(j,s) = 1$, we *L-certify* the computation and change $\Delta(x)$. This case can only occur finitely often, as usual.

It is what we do in the case that $g(j,t) = 0$ and $n \neq L_t \upharpoonright \varphi_{e,s}(x)$ that causes us problems. We will ensure that there is at most *one* $n \in V_s$ so that $n = L_s \upharpoonright m$ for some m at any stage s. We will ensure that V is a prefix free set. Suppose that we are in this case. There are two basic possibilities:

(i) Some $z \leq \varphi_{e,s}(x)$ *leaves* L after stage s. (We refer to this as "L moves right.")

This is the good case. The set L is d.c.e. and hence this z can never return. Thus "$g(j,s) = 0$" is also L-certified.

(ii) L moves left. That is, $L_t \upharpoonright \varphi_{e,s}(x) \supset L_s \upharpoonright \varphi_{e,s}(x)$.

This is the real problem. Now it might in the future be possible for $L_{s'} \upharpoonright \varphi_{e,s}(x) = L_s \upharpoonright \varphi_{e,s}(x)$.

The main idea is that should such a future stage s' occur, with $g(x,s') = 1$, we will be able to claim that we can make $A_{s'+1} \upharpoonright \varphi_{e,s}(x) = A_s \upharpoonright \varphi_{e,s}(x)$ by extraction of numbers from A. That is, we are claiming that since we have not been left of L_s, A_v will not have been left of A_s for any stage v with $s \leq v \leq s'$.

It is by no means clear that we will be able to so restore A, and this is the core of our construction. It would well seem that perhaps the first time we saw some potential stage to act, we had some p in A_s, and some other N_k might act before stage s' in the sense of the above, perhaps causing elements to leave A_v for some stage v with $s < v \leq s'$. (Recall that only N-type requirements extract elements from A.) If this *could* occur then we would have no hope of making A d.c.e. and still meeting the requirements. The section below is devoted to the analysis of two N-type requirements above an M-type one and showing that "timing" considerations make this scenario impossible.

2.4 Two N-strategies above one M-type requirement Suppose there are two requirements N_0 and N_1 above a requirement M_0. Suppose at the current stage s, $n \in V_{1,s'}$ for some $s' < s$ and n is an initial segment of $L_s \upharpoonright \varphi_{1,s'}(x_1)$ and $g(j_1, s) = 1$.

Thus, at stage s, N_1 desires to restore the computation at stage $s+1$ back to the stage s' configuration $\leq \varphi_1(x_1)[s']$. We would like to be able to pull all of the elements $z < \varphi_{1,s'}(x_1)$ which are in A_s but not in $A_{s'}$ out of A. We claim $A_s \upharpoonright \varphi_{1,s'}(x_1) \supseteq A_{s'} \upharpoonright \varphi_{1,s'}(x_1)$. Otherwise, there must exist some number $z < \varphi_{1,s'}(x_1)$ which was in $A_{s'}$ but was pulled out at a stage s'' between s' and s. Since we do not extract numbers for the action of M-type requirements, z must have been pulled out by N_0 or N_1.

(i) z was pulled out by N_1 itself. Then, inductively, we must have restored a computation at stage s'' to an earlier stage $t < s'$ (since $z \in A_{s'}$ but $z \notin A_{s''}$). Since L is d.c.e., there cannot be any number $y < \varphi_{1,t}$ removed from L_t between stage t and s''. Otherwise, the computation at stage t cannot be restored. But the computation at stage t was destroyed by L, so $L_t \upharpoonright \varphi_{1,t} \subset L_{s'} \upharpoonright \varphi_{1,t}$. Thus $L_{s''} \upharpoonright \varphi_{1,t} = L_t \upharpoonright \varphi_{1,t} \subset L_{s'} \upharpoonright \varphi_{1,t}$.

(ia) $\varphi_{1,t} \le \varphi_{1,s'}$. It means there is a number below $\varphi_{1,s'}$ which left L by stage s, and so the computation at stage s' cannot be restored, a contradiction.

(ib) $\varphi_{1,t} > \varphi_{1,s'}$. Note that, by the use principle, $L_{s'} \upharpoonright \varphi_{1,s'}$ cannot be an initial segment of $L_t \upharpoonright \varphi_{1,t}$. This means that there is some number p below $\varphi_{1,s'}$ which entered L *after* stage t and will still be in L at stage s, a contradiction, since the assumption is that N_1 wished to restore to the stage t configuration before stage s and this would therefore necessitate p not being in L_t and hence not in L_s as L is d.c.e.

(ii) z was pulled out by N_0. Then, inductively, we must have restored a computation of N_0 at stage s'' to an earlier stage $t < s'$ (since $z \in A_s$ but $z \notin A_{s''}$). Since L is d.c.e., there cannot be any number $y < \varphi_{0,t}$ leaving L_t between stage t and s''. Otherwise, the computation at stage t cannot be restored. But the computation at stage t was destroyed by L, so $L_t \upharpoonright \varphi_{0,t} \subset L_{s'} \upharpoonright \varphi_{0,t}$. Thus $L_{s''} \upharpoonright \varphi_{0,t} = L_t \upharpoonright \varphi_{0,t} \subset L_{s'} \upharpoonright \varphi_{0,t}$.

(iia) $\varphi_{0,t} \le \varphi_{1,s'}$. It means there is a number below $\varphi_{1,s'}$ which left L by stage s, and so the computation at stage s' cannot restored. The point here is that we did nothing for N_0 at stage s', and hence $L_t \upharpoonright \varphi_{0,t}$ must be right of $L_{s'} \upharpoonright \varphi_{0,t}$, a contradiction.

(iib) $\varphi_{0,t} > \varphi_{1,s'}$. Since L is d.c.e., there cannot be any number $y < \varphi_{1,s'}$ removed from $L_{s'}$ between stage s' and s. Otherwise, the computation at stage s' cannot be restored. But the computation $\Phi_{1,s'}^{A_{s'} \oplus L_{s'}}(x_1)$ was destroyed by L at stage $s'+1$. So $L_{s'} \upharpoonright \varphi_{1,s'} \subset L_{s''} \upharpoonright \varphi_{1,s'} = L_t \upharpoonright \varphi_{1,s'}$. This means there is some number below $\varphi_{1,s'}$ which entered L before stage s' and left L between stage s' and s''. So the computation $\Phi_{1,s'}^{A_{s'} \oplus L_{s'}}(x_1)$ cannot be restored, a contradiction.

We now turn to the formal details of this finite injury argument.

3 The Proof of Theorem 1.6

3.1 Basic module For M_e we build a c.e. set U whose index i is given by the Recursion Theorem.

1. Pick a large fresh follower m.
2. Wait for $A \upharpoonright m+1 = \Psi^L \upharpoonright m+1[s]$.
3. Put $n = L \upharpoonright \psi(m)[s]$ into U.
4. Run the enumerations of g and L until

   Case (1) $g(i,t) = 1$. Put m into A_{t+1}. Declare that m is used, and restrain it with priority M_e from leaving A.

   Case (2) $L \upharpoonright \psi(m)[t]$ changed. Go to step (2).

5. If $A \upharpoonright m+1 = \Psi^L \upharpoonright m+1[t]$ and m has been used go back to step (1).

For N_e, we build a c.e. set V. By the Recursion Theorem, we assume V has index j.

1. Pick up a fresh follower m.
2. Wait for $\Delta \upharpoonright m+1 = \Phi_e^{L \oplus A} \upharpoonright m+1[s]$, or L_s extends some $n' = L \upharpoonright \varphi(m)[t]$ already in V_s, and $g(j,s) = 1$.

3. In the case that n' is an initial segment of L_s, go to (5), Case (2), Subcase (1) below.
4. Put $n = L \upharpoonright \varphi(m)[s]$ into V.
5. Run the enumerations of g and L until

   Case (1) $g(j,t) = 1$. Put (or pull) m into (out of) Δ so that $\Delta(m) \neq \Phi_e^{L \oplus A}(m)$. Protect this computation by setting up a restriction r.

   Case (2) $L \upharpoonright \varphi(m)[t]$ changed.

   Subcase (1) $n' = L \upharpoonright \varphi(m)[t]$ was already put into V at some previous stage t' and $g(j,t) = 1$. Pull out the numbers which have entered A after stage t'. Put (or pull) m into (out of) Δ so that $\Delta(m) \neq \Phi_e^{L \oplus A}(m)[s]$. Go to step (2).

   Subcase (2) Otherwise. Go to step (2).

3.2 Construction Order the priorities of the requirements $M_0, N_0, M_1, \ldots$. For every requirement N_e, we build a c.e. set V_e and by the Recursion Theorem, it has an index $j(e)$, and similarly $U_e = W_{i(e)}$ for M_e. Every set we are constructing except A and Δ is a local set. We will differentiate between R requiring attention, and R *acting*. It will only be in the latter case that R will initialize lower priority requirements to preserve its action. Also we will use the phrase "speed up the L-enumeration to wait for. . . to occur." We will regard this as happening in one step of the construction, so that any action can be taken at the current stage. This avoids having stages where nothing is done while we are waiting for some pending decision for some requirement, and considerably simplifies the notation. We will use the L-enumeration given by this process within the stage. Thus, if some requirement receives attention and, while we are examining it, we see some new L-configuration consistent with some $g(i,t)$ or $g(j,t)$ because some number entered or left $L_t - L_s$, then we will regard this version of L, L_t, as replacing L_s, when the next requirement is considered. We will simply always denote the current version by L_s and will regard its meaning as being clear by context. Requirements will be considered in increasing order of priority at each stage to see if they require attention. At most one requirement can require attention and at most once per stage. We say a requirement R requires attention at stage $s+1$ if one of the following holds.

1. $R = M_e$

   Case (1) M_e has no unused follower, and the length of agreement between Ψ^L and A has just increased. Then our action is to appoint an (unused) follower m_s to M_e.

   Case (2) M_e currently has an unused follower assigned at some stage $t < s$, so that $m_{e,s} = m_{e,t}$, and $A_s \upharpoonright m_{e,t} + 1 = \Psi_{e,s}^{L_s} \upharpoonright m_{e,t} + 1$. (It will be the case that $A_s(m_{e,t}) = 0 = \Psi_{e,s}^{L_s}(m_{e,t})$.) Set $U_{e,s+1} = U_{e,s} \cup \{n\}$ where $n =_{\text{def}} L_s \upharpoonright \psi_{e,s}$. Speed up the L-enumeration L and $g(i(e), s)$ until a stage $s' > s$ so that either $L_{s'} \upharpoonright \psi_{e,s}(m_{e,t}) \neq L_s \upharpoonright \psi_{e,s}(m_{e,t})$ or $g(i(e), s') = 1$. In the former case, do nothing. If $g(i(e), s') = 1$ then declare that $m_{e,s}$ is used, and that M_e acts. Initialize lower priority requirements. Enumerate $m_{e,s}$ into A_{s+1}.

2. $R = N_e$.

   Case (1) N_e has no follower at stage s. Our action is to appoint a fresh number $m = m_{e,s}$ to follow N_e. Here N_e acts and we initialize all lower priority requirements.

   Case (2) N_e has a follower $m_{e,s}$ appointed and not canceled since some stage $t < s$, so that $m_{e,s} = m_{e,t}$. See if $\Delta_s(m_{e,t}) = \Phi_{e,s}^{L_s \oplus A_s}(m_{e,t})$, or L_s extends some $n' = L \restriction \varphi_{e,u}(m_{e,t})[u]$ already in $V_{e,s}$ and $g(j(e), s) = 1$.

   Subcase (2.0) No. Do nothing.

   Subcase (2.1) Yes and $n = L \restriction \varphi_e(m_{e,t})[s]$ does not extend some initial segment already in $V_{e,s}$. Speed up the enumeration of L and $g(j(e), s)$ until a stage $s' > s$ so that either $L_{s'} \restriction \varphi_{e,s}(m_{e,t}) + 1 \neq L_s \restriction \varphi_{e,s}(m_{e,t}) + 1$ or $g(j(e), s') = 1$. In the former case, we do nothing. If $g(j(e), s') = 1$ then N_e acts. The action is to initialize lower priority requirements, and make $\Delta_{s+1}(m_{e,t}) \neq \Phi_{e,s}^{L_s \oplus A_s}(m_{e,t})$.

   Subcase (2.2) Yes and $n = L \restriction \varphi_e(m_{e,t})[s]$ is compatible with, that is, L_s extends, some $n' = L \restriction \varphi_e(m_{e,t})[u]$ for some $t < u < s$ already in $V_{e,s}$. Speed up the enumeration of L and $g(j(e), s)$ until a stage $s' \geq s$ so that either $L_{s'} \restriction \varphi_{e,s}(m_{e,t}) + 1 \neq L_s \restriction \varphi_{e,s}(m_{e,t}) + 1$ or $g(j(e), s') = 1$. In the former case, we do nothing. If $g(j(e), s') = 1$ then N_e acts. The action is to initialize lower priority requirements, cause $A \restriction \varphi_{e,s}(m_{e,t})[s+1] = A_u \restriction \varphi_{e,u}(m_{e,t})$, and make $\Delta_{s+1}(m_{e,t}) \neq \Phi_{e,s}^{L_s \oplus A_s}(m_{e,t})$ (which will happen if we restore $\Delta_{s+1}(m_{e,t}) = \Delta_u(m_{e,t})$. (Naturally we will need to check that this can be done while making A d.c.e.)

Notice that the follower m_e for N_e will never be canceled once it is defined provided that N_e has priority (meaning that we have reached a stage where higher priority requirements have ceased to act on the construction).

3.3 Verification We prove that every requirement is satisfied and acts and is initialized at most finitely often, by induction on the priority f. Select the least stage s_0 so that all of the requirements of higher priority than R_f have ceased activity. Note if a requirement receives attention without acting then it will never put (or pull) anything into (out from) A and/or Δ. Suppose all of the following lemmas are true for every requirement of priority higher than R_f. Suppose R_f is M_e or N_e. Define $V_e = \cup_{t \geq s} V_{e,t}$ and $U_e = \cup_{t \geq s} U_{e,t}$.

Lemma 3.1 *For each M_e-requirement, U_e is prefix free, $A \neq \Psi_e^L$, and M_e acts only finitely often.*

Proof We work after the stage s_0 where M_e will never again be initialized. First, we prove U_e is a prefix free set. It suffices to prove $U_{e,s'}$ is prefix free for every $s' \geq s_0$. Suppose we put an initial segment $n_{s'} = L_{s'} \restriction \psi_{s'}(m_{e,s'})$ into $U_{e,s'+1}$ at stage $s'+1$. If $U_{e,s+1}$ were not prefix free, then there must a segment $n_t = L_t \restriction \psi_t(m_{e,t})$ compatible with $n_{s'}$ which has been put into $U_{e,t+1}$ at some stage $s_0 \leq t+1 < s'$.

1. $\psi_t(m_{e,t}) < \psi_{s'}(m_{e,s'})$ and $L_{s'} \restriction \psi_t(m_{e,t}) = L_t \restriction \psi_t(m_{e,t})$. If $m_{e,t} < m_{e,s'}$ then we will have acted for $m_{e,t}$ to have appointed $m_{e,s'}$, $A_{s'+1}(m_{e,t}) = A_{t+1}(m_{e,t}) = 1 \neq 0 = \Psi_{e,t}^{L_t}(m_{e,t}) = \Psi_{e,s'}^{L_{s'}}(m_{e,t})$. Hence M_e would not

have received attention at stage s', a contradiction. If $m_{e,t} = m_{e,s'}$, then by the basic properties of uses for reductions the corresponding n_t and $n_{s'}$ cannot be compatible.

2. Otherwise. Then $\psi_t(m_{e,t}) > \psi_{s'}(m_{e,s'})$ and $L_t \upharpoonright \psi_{s'}(m_{e,s'}) = L_t \upharpoonright \psi_{s'}(m_{e,s'})$. Again if we assume the two ns to be compatible, it can only be that $m_{e,s'} \neq m_{e,t}$, and so $m_{e,s'}$ was appointed *after* we acted for M_e using $m_{e,t}$. But then, $m_{e,s'}$ is appointed as a fresh number and we would have made $m_{e,s'}$ to exceed all previous uses seen in the construction at the stage u with $t + 1 \leq u < s'$ at which it was appointed. In particular, $m_{e,s'}$, and hence $\psi_{s'}(m_{e,s'})$, will exceed $\psi_t(m_{e,t})$, so this case cannot occur.

Now choose a stage $s' \geq s_0$ so that $\forall t \geq s'(g(i(e), t) = g(i(e), s'))$. There are two cases.

1. $g(i(e), s') = 0$. Select a stage $s'' \geq s$ so that a follower $m_{e,s''}$ has been defined and $A_{s''}(m_{e,s''}) = 0$. Then it will never be initialized and M_e will never require attention after s''. $U_e \cap \{L \upharpoonright n, n \in \mathbb{N}\} = \varnothing$. This means that $\Psi_e^L(m_{e,s''})$ is undefined or defined and $\neq 0$, and so M_e is satisfied and will never require attention.
2. $g(i(e), s') = 1$. Since U_e is prefix free, $|U_e \cap \{L \upharpoonright n, n \in \mathbb{N}\}| \leq 1$. So $|U_e \cap \{L \upharpoonright n, n \in \mathbb{N}\}| = 1$. Select a stage $s'' \geq s'$ so that $|U_{e,s''} \cap \{L_t \upharpoonright n, n \in \mathbb{N}\}| = 1$ for every stage $t \geq s''$. Then there must be one follower $m \leq |U_e \cap \{L \upharpoonright n, n \in \mathbb{N}\}|$ so that $\Psi^L(m) = 0$. So M_e must have acted to put m into A. So the requirement will never require attention after stage s''.

$\square$

Lemma 3.2 *For each N_e requirement, V_e is prefix free, $\Delta \neq \Phi_e^{L \oplus A}$ and N_e requires attention only finitely often.*

Proof Again we work at stages $s \geq s_0$ after which V_e is initialized for the last time. In the following we are proving the lemma above, but not proving here that A is d.c.e. The construction for N_e only asks us to restore A to earlier configurations. We will in a subsequent lemma ensure that such restorations are possible while still keeping A d.c.e. First, we prove V_e is prefix free. Suppose there is a follower $m_{e,s}$ at stage s, and this is the least stage after s_0 when N_e picks a follower. Then, by construction, this follower is immortal. Let $m_e = m_{e,s}$. Notice that this follower and any activity after stage s_0 cannot affect anything of higher priority than N_e as the numbers involved are too big by initialization.

It suffices to prove $V_{e,s'}$ is prefix free for every $s' \geq s$. Suppose we put a number $n_{s'} = L_{s'} \upharpoonright \varphi_{s'}(m_e)$ into $V_{e,s'+1}$ at stage $s' + 1$. If $V_{e,s+1}$ were not prefix free, then inductively, if s' is the least stage where it becomes nonprefix free, there must a number $n_t = L_t \upharpoonright \varphi_t(m_e)$ which has been put into $U_{e,t+1}$ at some stage $s \leq t + 1 < s'$ and this initial segment is compatible with $n_{s'}$.

At the stage $t+1$ when we enumerated n_t into V_e, we would have had a computation $\Phi_{e,t}^{L_t \oplus A_t}(m_e) \downarrow = \Delta_t(m_e)$. The way the construction works is that we would *not* add $n_{s'}$ to V_e. (Rather we would check in Subcase (2.2) to see if we can L-certifiably via $g(j(e), s')$ restore A to force a disagreement. Thus this cannot occur.) So V_e must be prefix free.

Now select the least stage $s'' \geq s_0$ so that $\forall t \geq s''(g(j(e), t) = g(j(e), s''))$.

1. $g(j(e), s'') = 0$. Then N_e will never require attention after s''. By the Recursion Theorem, $V_e \cap \{L \upharpoonright n, n \in \mathbb{N}\} = \varnothing$. It is immediate that $\Phi_e^{L \oplus A}(m_e) \neq \Delta(m_e)$
2. $g(j(e), s'') = 1$. Since V_e is prefix free, $|V_e \cap \{L \upharpoonright n, n \in \mathbb{N}\}| \leq 1$, and since $\lim_s g(j(e), s) = 1$, we see $|V_e \cap \{L \upharpoonright n, n \in \mathbb{N}\}| = 1$. Select the least stage, say t, $t \geq s$ at which we put some the $n_e = L_t \upharpoonright \varphi_{e,t}(m_e) \in \{L \upharpoonright m, m \in \mathbb{N}\}$ into $V_{e,t}$. Now this is a real initial segment of L and it is in V_e at every stage after t. There is some least stage $t' \geq t, s''$ such that $L_u \upharpoonright \varphi_{e,t}(m_e)[u] = n_e$ for all $u \geq t'$. At such a stage u, we will see that Subcase (2.2) (or (2.1)) pertains as we have L extending n_e. At such a stage t' we would restore $A_t \upharpoonright \varphi_{e,t}(m_e)[t'] = A_t \upharpoonright \varphi_{e,t}(m_e)[t]$, since, by choice of $t' \geq s''$, $g(j(e), t') = 1$. This would create a disagreement which would be preserved forever, and N_e would never again act. □

Lemma 3.3 *A is d.c.e.*

Proof If this is not true then there is some number x which enters and leaves more than once. The only requirements that act in a non-c.e. manner upon A are the N_es. It follows that there must be two requirements N_a and N_b where N_a acts to take a number out of A and N_b acts to put it back in. Since N_b acts after N_a's action, it must have higher priority than N_a, as it was not initialized by N_a's action (in which case its numbers would be too big). Now since N_a acts to take x out of A at some stage s_a, it can only do so through the action of Subcase (2.2) of the construction. That is, we must have seen L_{s_a} extending some n_a in V_a. This n_a was put into V_a at some stage $t_a < s_a$. At this stage we would have had an apparent $\Phi_a^{L \oplus A}(m_a) = \Delta(m_a)$ computation not corresponding to some earlier configuration in V_a. It must have been that $x \notin A_{t_a}$. Thus x must have entered A at some stage v after stage t_a and this can only happen through the action of some M_k of lower priority than both N_a and N_b since M_k did not initialize them. Since M_k has lower priority and x is smaller than $\varphi_a(m_a)$, it must have been that N_a did not act at any stage before v, since otherwise x would have been too big. (We remark that here and below, no M-type requirement can ever re-enumerate a number since whenever they once enumerate it, if that number is then extracted by some N that N must have had *higher* priority, initializing M, causing it to pick a fresh number to follow it.) We can only conclude that for all stages between t_a and v, $L_v \upharpoonright \varphi_a(m_a)[t_a] \supset L_{t_a}\varphi_a(m_a)[t_a]$. (It cannot have moved right else it could not ever get back to L_u extending $L_{t_a} \upharpoonright \varphi_a(m_a)[t_a]$.)

Now if N_b acts to put x back into A, then since it restores A to a configuration corresponding to a $\Phi_b^{L \oplus A}(m_b)[q]$, it must have been that x was already in A_q. Since this configuration must occur before the stage s_a where N_a acts to take x out of A, we must conclude that $L_q \upharpoonright \varphi_a(m_a)[t_a] \supset L_{t_a} \upharpoonright \varphi(m_a)$.

But now we have a contradiction. For N_a to act before N_b acts, we would need that L moves right so that L_{s_a} extends $L_{t_a} \upharpoonright \varphi_a(m_a)[t_a]$. However, since N_b does not act before stage s_a (lest it initialize N_a), we must have that $g(j(b), s) = 0$ for stages $q \leq s \leq s_a$, and this must be L-certified in the sense that L_s cannot be compatible with $L_q \upharpoonright \varphi_b(m_a)[q]$. (Otherwise Subcase (2.2) would apply to N_b.) Since $s = s_a$ is a special case of this noncompatibility, L_{s_a} cannot be compatible with $L_q \upharpoonright \varphi_b(m_b)[q]$. The conclusion is that $L_{s_a} \upharpoonright \varphi_b(m_b)[q]$ is left of $L_q \upharpoonright \varphi_b(m_b)[q]$.

This is only possible if $\varphi_b(m_b)[q] > \varphi_a(m_a)[t_a]$. But finally we have a contradiction. $L_{s_a} \restriction \varphi_a(m_a)[t_a]$ is right of $L_{t_a} \restriction \varphi_a(m_a)[t_a]$. □

Lemma 3.4 Δ *is* Δ^0_2.

Proof It suffices to prove for every uncanceled follower m_e and $e \in \mathbb{N}$, $\Delta(m_e) \downarrow$. But every requirement requires attention at most finitely often. So it is true. □

4 Some Comments

It is not difficult to modify the construction above to prove the following.

Corollary 4.1 *For any low d.c.e set L, there is a low d.c.e set A with* $L <_T A$.

Proof We can replace the N_e of Theorem 1.6 by standard lowness requirements

$$\exists^\infty s(\Phi_e^{A\oplus L}(e)[s] \downarrow) \to \Phi_e^{A\oplus L}(e) \downarrow .$$

Again the argument is finite injury. Now *after* stage $s = s(e)$ where N_e is initialized for the last time by higher priority requirements, N_e has the ability to restore A at will to any configuration involving $A \restriction \varphi_u(e)[u]$ for stages $s(e) \leq u$, and the ability to protect such restorations by initializing lower priority requirements. Thus we will use V_e and g in the same way after $s(e)$ (the final incarnation of V_e), to either see L_s extending something already in V_e which is g-certified ($g(e, s) = 1$) or we see a new $\Phi_e^{A\oplus L}(e)[u]$ computation, which we will L-test via V_e and g. Again, by the definition of g in the theorem and the proof, if $\lim_s g(e, s) = 1$ then $\Phi_e^{A\oplus L}(e) \downarrow$ and if $\lim_s g(e, s) = 0$ then $\Phi_e^{A\oplus L}(e) \uparrow$. So $A \oplus L$ is low. □

Corollary 4.2 *For any low d.c.e degrees, there are infinitely many d.c.e. degrees above it.*

Proof By Corollary 4.1. □

Now we finish with some brief remarks about extensions to our results. We think it is not hard (but tedious) to show that our argument works for L is n-c.e. ($\omega \geq n > 1$), using a nonuniform proof. We have not checked this in detail.

We want to explain why our argument does not work to solve Question 1.7. In our proof we can ensure A to be d.c.e. since no other requirements force A to change more often. However, in Question 1.7, we failed to construct such a d.c.e. set since we must put the numbers into A_{1-i} while we pull them out from A_i. This can happen many (although finitely many) times for a fixed number. This is the crucial difference between a Friedberg strategy and a Sacks strategy. Although we do not know whether Question 1.7 has a positive solution in the d.c.e. degrees, this method can be used to split every c.e. degree into two ω-c.e. degrees over any lesser low d.c.e. degree. The trick is that we can bound the times at which both A_is change by a computable function.[4]

Finally we remark that Kučera has observed that there are some limitations on the interactions of the c.e. degrees and the low degrees. He observed that by Theorem 2 of Kučera [6] in relativized form combined with the low basis theorem, we can show that given any low degree $\mathbf{d}$ there is a low PA degree $\mathbf{p} > \mathbf{d}$. Then by his priority-free solution to Post's problem, there is a c.e. nonzero degree $\mathbf{a} < \mathbf{p}$. The upshot is that $\mathbf{a} \cup \mathbf{d}$ is therefore low.

Notes

1. There is currently no known elementary difference between the low_2 d.c.e. degrees and the low_2 c.e. degrees.

2. Arslanov, Cooper, and Li have announced an affirmative answer to a variation of this question. Namely, we can split A into A_i of d.c.e. degree so that $A_0 \oplus L | T A_1 \oplus L$. Their proof uses different methods again and filters through a theorem of Arslanov, LaForte, and Slaman [3].

3. Since our construction is a finite injury argument, it will suffice to simply initialize lower priority requirements for "restraint" as lower priority requirements will then need to work with "fresh" numbers for their followers, which will be bigger than any seen in the construction before in the usual method for modern presentations of finite injury arguments.

4. This is more or less the method used by Arslanov, Cooper, and Li in their solution to the generalized version of 1.7, and *additionally* they arrange things so that you can also use the Arslanov, LaForte, Slaman Theorem [3].

References

[1] Arslanov, M. M., "Lattice properties of the degrees below $\mathbf{0}'$," *Doklady Akademii Nauk SSSR*, vol. 283 (1985), pp. 270–73. Zbl 0596.03040. MR 87e:03095. 147

[2] Arslanov, M., S. B. Cooper, and A. Li, "There is no low maximal d.c.e. degree," *Mathematical Logic Quarterly*, vol. 46 (2000), pp. 409–416. Zbl 0967.03037. MR 2001e:03076. 148

[3] Arslanov, M. M., G. L. LaForte, and T. A. Slaman, "Relative enumerability in the difference hierarchy," *The Journal of Symbolic Logic*, vol. 63 (1998), pp. 411–20. Zbl 0911.03021. MR 99f:03057. 158

[4] Cooper, S. B., L. Harrington, A. H. Lachlan, S. Lempp, and R. I. Soare, "The d.r.e. degrees are not dense," *Annals of Pure and Applied Logic*, vol. 55 (1991), pp. 125–51. Zbl 0756.03020. MR 93a:03045. 147

[5] Downey, R., "D.r.e. degrees and the nondiamond theorem," *The Bulletin of the London Mathematical Society*, vol. 21 (1989), pp. 43–50. Zbl 0628.03030. MR 90j:03082. 147

[6] Kučera, A., "On the use of diagonally nonrecursive functions," pp. 219–39 in *Logic Colloquium '87 (Granada, 1987)*, vol. 129 of *Studies in Logic and the Foundations of Mathematics*, North-Holland, Amsterdam, 1989. Zbl 0683.03024. MR 91c:03037. 157

[7] Lachlan, A. H., "A recursively enumerable degree which will not split over all lesser ones," *Annals of Mathematical Logic*, vol. 9 (1976), pp. 307–65. Zbl 0357.02040. MR 53:12912. 148, 149

[8] Robinson, R. W., "Interpolation and embedding in the recursively enumerable degrees," *Annals of Mathematics (2)*, vol. 93 (1971), pp. 285–314. Zbl 0259.02033. MR 43:51. 148

[9] Sacks, G. E., "The recursively enumerable degrees are dense," *Annals of Mathematics (2)*, vol. 80 (1964), pp. 300–312. Zbl 0135.00702. MR 29:3367. 147

Acknowledgments

The first author is supported by the Marsden Fund of New Zealand. The second author is supported by NSF of China No. 60310213, 19931020 and a postdoctoral fellowship from the New Zealand Institute for Mathematics and its Applications (Centre of Research Excellence) under the *2004 Logic and Computation Programme*.

School of Mathematics and Computing Sciences
Victoria University of Wellington
Wellington
New Zealand
Rod.Downey@mcs.vuw.ac.nz
yuliang@mcs.vuw.ac.nz