Nonexistence of Minimal Pairs in L[d]

Fang Chengling[1], Liu Jiang[2], Wu Guohua[3(✉)], and Mars M. Yamaleev[4]

[1] School of Science, Chongqing Jiaotong University, Chongqing, China
[2] Chongqing Institute of Green Intelligent Technology, Chinese Academy of Sciences, Chongqing, China
[3] School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore, Singapore
guohua@ntu.edu.sg
[4] Institute of Mathematics and Mechanics, Kazan Federal University, Kazan, Russia

Abstract. For a d.c.e. set D with a d.c.e. approximation $\{D_s\}_{s\in\omega}$, the Lachlan set of D is defined as $L(D) = \{s : \exists x \in D_s - D_{s-1} \text{ and } x \notin D\}$. For a d.c.e. degree $\mathbf{d}$, $L[\mathbf{d}]$ is defined as the class of c.e. degrees of those Lachlan sets of d.c.e. sets in $\mathbf{d}$. In this paper, we prove that for any proper d.c.e. degree $\mathbf{d}$, no two elements in $L[\mathbf{d}]$ can form a minimal pair. This result gives another solution to Ishmukhametov's problem, which asks whether for any proper d.c.e. degree $\mathbf{d}$, $L[\mathbf{d}]$ always has a minimal element. A negative answer to this question was first given by Fang, Wu and Yamaleev in 2013.

1 Introduction

For a d.c.e. set D with a d.c.e. approximation $\{D_s\}_{s\in\omega}$, the Lachlan set of D is defined as $L(D) = \{s : \exists x \in D_s - D_{s-1} \text{ and } x \notin D\}$.[1] Note that $L(D)$ is

Fang is partially supported by NSF of China (No. 11401061), SRF for ROCS, SEM and Chongqing Jiaotong University Fund (No. 2012kjc2-018).
Liu is partially supported by NSF of China (No. 61202131), the CAS western light program, Chongqing Natural Science Foundation (No. cstc2014jcsfglyjs0005 and No. cstc2014zktjccxyyB0031).
Wu is partially supported by a grant MOE2011-T2-1-071 (ARC 17/11, M45110030) from Ministry of Education of Singapore and by a grant RG29/14, M4011274 from NTU.
Yamaleev is partially supported by The President grant of Russian Federation (project NSh-941.2014.1), by Russian Foundation for Basic Research (projects 14-01-31200, 15-01-08252), by the subsidy allocated to Kazan Federal University for the project part of the state assignment in the sphere of scientific activities, and by a grant MOE2011-T2-1-071 (ARC 17/11, M45110030) from Ministry of Education of Singapore.

[1] This definition is from Ishmukhametov's articles [2] and [3]. Another definition of the Lachlan set is $L^*(D) = \{\langle x, s\rangle : x \in D_s \text{ and } x \notin D\}$. It is easy to see that $L(D)$ defined by Ishmukhametov and $L^*(D)$ above are Turing equivalent, and hence, make no difference when we consider $L[\mathbf{d}]$, a collection of Turing degrees. In this paper, we will use Ishmukhametov's definition.

A. Beckmann et al. (Eds.): CiE 2015, LNCS 9136, pp. 177–185, 2015.
DOI: 10.1007/978-3-319-20028-6_18

a c.e. set with $L(D) \leq_T D$ and that D is c.e. in $L(D)$. In [2], Ishmukhametov showed that the Turing degree of $L(D)$ doesn't depend on the approximations of D, and that for any X, $L(D) \leq_T X$ if and only if $D \in \Sigma_1^X$. In the same work, Ishmukhametov considered the following class of c.e. predecessors of d.c.e. degree $\mathbf{d}$,

$$R[\mathbf{d}] = \{\deg(W) : W \text{ is c.e. and there exists a } D \in \mathbf{d} \text{ such that } D \in \Sigma_1^W\},$$

and constructed a d.c.e. degree $\mathbf{d}$ such that $R[\mathbf{d}]$ has a minimum element. In [2], Ishmukhametov asked whether $R[\mathbf{d}]$ can always have a minimal element, if $\mathbf{d}$ is proper d.c.e. In [3], Ishmukhametov proved the existence of a d.c.e. degree such that the corresponding Lachlan degrees have no minimum element.

Theorem 1. *(Ishmukhametov [3]) There exists a proper d.c.e. degree* $\mathbf{d}$ *such that for any d.c.e. set* $B \in \mathbf{d}$ *there exists a d.c.e. set* $A \in \mathbf{d}$ *such that* $L(B) \not\leq_T L(A)$.

In [1], Fang, Wu and Yamaleev defined $L[\mathbf{d}]$ as the class of c.e. degrees of those Lachlan sets of d.c.e. sets in $\mathbf{d}$ and also proved that $R[\mathbf{d}] = L[\mathbf{d}]$. After this, Fang, Wu and Yamaleev constructed a proper d.c.e. degree $\mathbf{d}$ such that $R[\mathbf{d}]$ has no minimal element, providing a negative answer to Ishmukhametov's question.

Theorem 2. *(Fang, Wu, and Yamaleev [1]) There exists a proper d.c.e. degree* $\mathbf{d}$ *such that for any d.c.e. set* $B \in \mathbf{d}$ *there exists d.c.e. set* $A \in \mathbf{d}$ *such that* $L(A) <_T L(B)$.

In this paper, we will prove that in $L[\mathbf{d}]$, no two elements can form a minimal pair, if $\mathbf{d}$ is proper d.c.e.

Theorem 3. *If* $\mathbf{d}$ *is a d.c.e. degree then for any d.c.e. sets* $A, B \in \mathbf{d}$*, there exists a d.c.e. set* $D \in \mathbf{d}$ *such that* $L(D) \leq_T L(A), L(B)$*. In particular, if* $\mathbf{d}$ *is proper d.c.e., then no two elements in* $L[\mathbf{d}]$ *can have infimum* $\mathbf{0}$.

Combining Theorems 1 and 3, we can have an alternative proof of Theorem 2, and hence another solution of Ishmukhametov's question via Ishmukhametov's original attempt. Indeed, take $\mathbf{d}$ as constructed in Theorem 1, and assume that $B \in \mathbf{d}$ is a d.c.e. set such that $L(B)$ is a minimal element in $L[\mathbf{d}]$. Then by Theorem 1, there is a d.c.e. set A in $\mathbf{d}$ such that $L(B) \not\leq_T L(A)$. By Theorem 3, there is a d.c.e. set $D \in \mathbf{d}$ such that $L(D) \leq_T L(A), L(B)$ and $L(D)$ incomputable, and hence $L(D) <_T L(B)$. A contradiction.

Our notation is standard and generally follows Soare's textbook [4].

2 Basic Idea of the Construction

We now give some basic idea of the proof of Theorem 3. Let A, B be two d.c.e. sets with $A \equiv_T B$, and let $\{A_s\}_{s\in\omega}$, $\{B_s\}_{s\in\omega}$ be the d.c.e approximations of A and B. We will construct a d.c.e. set $D \equiv_T A$ (hence $D \equiv_T B$) such that $L(D) \leq_T L(A), L(B)$. Assume that $A = \Phi^B$ and $B = \Psi^A$ and we will build

partial computable functionals Γ, Δ such that $A = \Gamma^D$ and $D = \Delta^A$, and also partial computable functionals Λ, Θ such that $L(D) = \Lambda^{L(A)}$ and $L(D) = \Theta^{L(B)}$.

Construction of Γ and Δ. For each x, we reserve $2x$ and $2x+1$ to code A in D via a partial computable functional Γ. We define the γ-use $\gamma(x)$ as $\gamma(x) = 2x+1$.

When $\Gamma^D(x)$ is defined and $\Gamma^D(x) \neq A(x)$, then we enumerate $2x$ into D (we can always do so, if it is the first time for $A(x)$ to change, after $\Gamma^D(x)$ is first defined), and redefine $\Gamma^D(x) = A(x)$. If $A(x)$ changes again (the last change), we can enumerate $2x+1$ into D, to undefine it, or remove $2x$ from D to recover $\Gamma^D(x)$ (as 0). *Of course, when we have $\Gamma^D(x) \neq A(x)$ for the first time with $\Gamma^D(x) = 1 \neq 0 = A(x)$ (i.e. $A(x)$ has its first change before $\Gamma^D(x)$ is defined), then the enumeration of $2x$ into D will undefine $\Gamma^D(x)$ (we will then redefine it as 0) and as $A(x)$ will not change later, there is no possibility of removing $2x$ out of D later, to rectify $\Gamma^D(x)$.*

We also define Δ, to make $D(2x) = \Delta^A(2x)$ and $D(2x+1) = \Delta^A(2x+1)$ with use $\delta(2x) = \delta(2x+1) = \psi(\varphi(x))$. Of course, $\delta(2x)$ and $\delta(2x+1)$ should change accordingly, if $A\lceil\psi(\varphi(x))$ changes ($B\lceil\varphi(x)$ may have changes and hence $\varphi(x)$ may be increased), and there will be no such change eventually, as $\Phi^B(x)$ and $\Psi^A(\varphi(x))$ are assumed to be convergent computations.

Removing $2x$ can also happen because of the definition of Δ. It can happen that after the second change of $A(x)$ (as indicated above, $2x$ is enumerated into D, at stage s say, when $A(x)$ has its first change), $A\lceil\psi(\varphi(x))$ changes back to a previous configuration, and hence, according to the definition of Δ^A and to make $D(2x) = \Delta^A(2x)$, we need to keep $D(2x) = 0$, and such a removal of $2x$ from D actually enumerates s, i.e., $s^D(2x)$, into $L(D)$, this enumeration needs to get permissions from $L(A)$ and $L(B)$.

Construction of Λ and Θ. We now describe the construction of Λ and Θ which are built to make $L(D) = \Lambda^{L(A)} = \Theta^{L(B)}$.

Without loss of generality, we assume that x enters A after s_0, the stage when $\Gamma^D(x)$ is defined, and B changes below $\varphi(x)[s_0]$, say at a number y. If y leaves B, then $A_t\lceil\delta(2x)[s_0] \neq A_{s_0}\lceil\delta(2x)[s_0]$ for all $t > s^A(x)$, and $2x$ remains in D forever. So, we assume that y enters B. Note that $s^A(x), s^B(y) > s_0$ and that we put $2x$ into D, with $s^D(2x) > s^A(x), s^B(y)$. We let $s_1 = s^D(2x)$ and define $\Lambda^{L(A)}(s_1)[s_1] = \Theta^{L(B)}(s_1)[s_1]$, with $\lambda(s_1)[s_1] = \theta(s_1)[s_1] = s_1$ (both are bigger than $s^A(x)$ and $s^B(y)$, and it could happen that later, because of the changes of $L(A)$ and $L(B)$, they can become different from each other).

If later x leaves A, say at stage s_2, and $A_{s_2}\lceil\psi(\varphi(x))[s_0] = A_{s_0}\lceil\psi(\varphi(x))[s_0]$, then as $\Delta^A(2x)[s_2] = \Delta^A(2x)[s_0] = 0$, to make $D(2x) = \Delta^A(2x)$, we need to extract $2x$ from D, which means that we need permissions from both $L(A)$ and $L(B)$. A permission from $L(A)$ is just provided, because $s^A(x)$ is enumerated into $L(A)$ at stage s_2. However, if y does not leave B before stage s_2, then it must happen at stage s_2 since $\Psi^A(y)[s_2] = \Psi^A(y)[s_0] = 0$, and hence $s^B(y)$ is enumerated into $L(B)$ at this stage, undefining $\Theta^{L(B)}(s^D(2x))$. Thus, as a consequence, we can extract $2x$ out of D.

The case that y leaves B before stage s_2 is a bit more complicated, as before stage s_2, we may already redefine $\Theta^{L(B)}(s^D(2x))$, and at stage s_2, we have

$\Lambda^{L(A)}(s^D(2x))$ undefined but $\Theta^{L(B)}(s^D(2x))$ defined, and we could not extract $2x$ from D, and in this case, at stage s_2, we put $2x+1$ into D, to code that x leaves A.

Suppose that x leaves A at stage s_2, also assume that B has changes below $\varphi(x)[s_0]$ between stages s_0 and s_2. Then A has changes below $\psi(\varphi(x))[s_0]$. We can assume that some y' enters B and that x' enters A between s_0 and s_2. Recall that at stage s_2, we put $2x+1$ into D to code that x leaves A, and also we redefine

$$\delta(2x)[s_2] = \delta(2x+1)[s_2] = \max_{s_0 \le t \le s_2} \{\psi(\varphi(x))[t]\}.$$

As $s^A(x) < s_1$ enters $L(A)$, $\Lambda^{L(A)}(s_1)$ is undefined at stage s_2 and we can refine

$$\lambda(s_1)[s_2] = \lambda(s_2)[s_2] = s_2.$$

Note that when y leaves B (before stage s_2), $\Theta^{L(B)}(s_1)$ is undefined (as $s^B(y)$ enters $L(B)$), and we would have already redefined $\Theta^{L(B)}(s_1)$ with

$$\theta(s_1)[s_2] = \theta(s_2)[s_2] = s_1' > s_1.$$

In particular, $s_2 > s^A(x')$ and that $s_1' > s^B(y')$. This kind of enumerations can be repeated several times, and stop eventually because Φ^B and Ψ^A are assumed to be total. So, without loss of generality, we assume that x', y' are the last numbers in this sequence.

Now suppose that at stage $s_3 > s_2$, $A_{s_3}\lceil\psi(\varphi(x))[s_0] = A_{s_0}\lceil\psi(\varphi(x))[s_0]$, which means that $\Lambda^A(2x)$ should be 0, and to make $D = \Lambda^A$, we need to extract $2x$ and $2x+1$ from D. *Of course, if* $A\lceil\psi(\varphi(x))[s_0]$ *does not recover to* $A_{s_0}\lceil\psi(\varphi(x))[s_0]$, *then we can just leave* $2x$ *and* $2x+1$ *in* D. To extract $2x$ and $2x+1$ from D at stage s_3, we must have permissions from $L(A)$ and $L(B)$, to enumerate s_1, s_2 into $L(D)$. Indeed, as x' leaves A, $s^A(x') \le s_2$ is enumerated into $L(A)$, which allows to correct $\Lambda^{L(A)}$ at both s_1, s_2. As $B = \Psi^A$, and $A_{s_3}\lceil\psi(\varphi(x))[s_0] = A_{s_0}\lceil\psi(\varphi(x))[s_0]$, $B(y')[s_3] = B(y')[s_0] = 0$, which means that y' also leaves B at stage s_3, and hence $s^B(y') < s_1'$ enters $L(B)$ at stage s_3, an $L(B)$-permission for s_1 and s_2.

3 Construction

First, we define expansionary stages for $A = \Phi^B$ and $B = \Psi^A$ in a standard way. Namely, we define the length agreement as

$$\ell(s) = \max\{y < s : \forall x < y\ [\Phi^B(x)[s] \downarrow = A(x)[s]\ \&\ \forall u < \varphi_s(x)(\Psi^A(u)[s] \downarrow = B(u)[s])]\}.$$

A stage s is expansionary if $s = 0$ or $\ell(s) > \ell(t)$ for all $t < s$.

We consider each stage of the form $4n$ as an expansionary stage, and we call these stage working stages. By this convention, between two stages $4n$ and $4(n+1)$, A and B can have many changes on numbers below $4n$. We will use $\mathbf{4n}$ to denote the n-th working stage. So between two working stages $4\mathbf{n}$ and $4(\mathbf{n+1})$, there are many stages of enumerations of A and B, and we assume

that at each stage, there is at most one x and at most one y, such that $A(x)$ and $B(y)$ change the value. We need this convention, as we need the definition of s^A and s^B well-defined. We may interpret this idea as, after working stage $4\mathbf{n}$, after stage $4\mathbf{n}+3$, at a stage t, we check whether t is an expansionary stage, and if yes, then t is the next working stage, i.e. $4(\mathbf{n}+\mathbf{1})$, and if no, then do nothing and go to stage $t+1$.

We use $\alpha(x)$ to denote the working stage $\mathbf{s}$ at which $\ell(\mathbf{s})$ exceeds x for the first time. We will ensure that at each stage, at most one number is enumerated into D (actually, only at stages $4\mathbf{n}+1, 4\mathbf{n}+2, 4\mathbf{n}+3$), and thus each stage s, at most one x enters D, and hence different x will have different $s^D(x)$. We clarify this as we are constructing reductions Λ, Θ, reducing $L(D)$ to $L(A)$ and $L(B)$ respectively. We will call $[4\mathbf{n}, 4\mathbf{n}+3]$ the n-th working block.

Working Block $[\mathbf{0}, \mathbf{3}]$

Let $D_0 = \emptyset$, and initiate all the functionals being constructed.

Working Block $[\mathbf{4s}, \mathbf{4s+3}]$

Stage $4\mathbf{s}$**.** Stage $4\mathbf{s}$ is an expansionary stage and at this stage, A and B may have many changes below $4\mathbf{s}$.

1. Let $x_1 < \ell(4\mathbf{s})$ be the least number being enumerated into A at stage $4\mathbf{s}$, if any.
2. Let $x_2 < \ell(4\mathbf{s})$ be the least number being removed from A at stage $4\mathbf{s}$, if any.
3. Check whether there exist some $z < \mathbf{s}$ with $D(2z)[4\mathbf{s}] = 1$ and $\Delta^A(2z)[4\mathbf{s}] = 0$, if any.

Go to next stage.

Stage $4\mathbf{s}+1$. Check whether $\Gamma^D(x_1)[4\mathbf{s}]$ is defined or not. If $\Gamma^D(x_1)[4\mathbf{s}]$ is defined, then enumerate $2x_1$ into D. Otherwise do nothing.

(*Thus* $4\mathbf{s}+1$ *is defined as* $s^D(2x_1)$. *The enumeration of* $2x_1$ *undefines* $\Gamma^D(x_1)$, *so we can redefined it as 1 later. It can happen that there are* x' *with* $x_1 < x' < \ell(4\mathbf{s})$ *that enters* A *at stage* $4\mathbf{s}$, *and the enumeration of* $2x_1$ *into* D *at stage* $4\mathbf{s}+1$ *also undefines* $\Gamma^D(x')$. *We may remove* $2x_1$ *from* D *later, at some stage* $4\mathbf{s}'+2$ *say, then at the same stage, we need to enumerate* $2x'$ *into* D, *to make sure that* $\Gamma^D(x')$ *does not recover to* $\Gamma^D(x')[4\mathbf{s}]$. *Because of this, we call* x_1 *the support of those numbers* $x' \geq x_1$, *which are also enumerated into* A *at stage* $4\mathbf{s}$.)

Go to next stage.

Stage $4\mathbf{s}+2$. Check whether $2x_2$ has been enumerated into D before stage $4\mathbf{s}$.

If not, then enumerate $2x_2$ into D to undefine $\Gamma^D(x_2)$. ($4\mathbf{s}+2$ *is defined as* $s^D(2x_2)$. *In this case, when we redefine* $\Gamma^D(x_2)$ *after stage* $4\mathbf{s}+2$, *we will define* $\Gamma^D(x_2)$ as 0, *and as* A *is d.c.e.,* $A(x_2)$ *will not have further changes. Thus,* $2x_2$ *will remain in* D *and* $s^D(2x_2)$ *cannot be enumerated into* $L(D)$.)

If $2x_2$ has been enumerated into D before stage $4\mathbf{s}$, then we check whether there is a working stage t with $\alpha(x_2) < t < s^A(x_2) < 4\mathbf{s}$ such that

$$A[4\mathbf{s}]\lceil\delta(2x_2)[t] = A[t]\lceil\delta(2x_2)[t].$$

Here $\alpha(x_2)$ is the first working stage s_0 at which $\ell(s_0) > x_2$.

– If there is such a t, then remove $2x_2$ from D. We also check whether there is some x' with x_2 as a support, and x' is still in $A[4\mathbf{s}]$. If such an x' exists, then we choose x' as the least one, and enumerate $2x'$ into D. For those numbers with x_2 as support before, we now use x' as their support.
  ($4\mathbf{s}+2$ *is defined as* $s^D(2x')$. *As* $2x_2$ *is removed from* D, $s^D(2x_2)$ *is enumerated into* $L(D)$. *Note that the enumeration of* $s^D(2x_2)$ *into* $L(D)$ *is permitted by* $L(A)$ *and* $L(B)$, *by simple argument.*)
– If there is no such a working stage t, then enumerate $2x_2 + 1$ into D (hence $\Gamma^D(x_2)$ *is undefined, and also note that both* $\Delta^A(2x_2)$ *and* $\Delta^A(2x_2 + 1)$ *are undefined at this stage, because of our assumption and that* x_2 *leaves* A, *and this allows us to enumerate* $2x_2 + 1$ *into* D).

Go to next stage.

Stage $4\mathbf{s} + 3$. Remove all these $2z$ out of D, together with $2z + 1$, if the associated $2z + 1$ is also in D. Let x_3 be the least number that has one of these z as its support. Enumerate $2x_3$ into D.

(*These numbers* $2z$ (*also* $2z+1$, *if in* D *already*) *have to be removed from* D, *as we need to keep* $D = \Delta^A$.)

Extending Definitions of $\Gamma, \Delta, \Lambda, \Theta$:

Γ: Find the least x such that $\Gamma^D(x)[4\mathbf{s} + 3]$ is not defined, and define $\Gamma^D(x) = A(x)[4\mathbf{s} + 3]$ with use $\gamma(x) = 2x + 1$.

Δ: Find the least x such that $\Delta^D(2x)[4\mathbf{s} + 3]$ is not defined, and define $\Delta^A(2x)[4\mathbf{s}+3] = D(2x)[4\mathbf{s}+3]$, $\Delta^A(2x+1)[4\mathbf{s}+3] = D(2x+1)[4\mathbf{s}+3]$, with use $\delta(2x)[4\mathbf{s} + 3] = \delta(2x + 1)[4\mathbf{s} + 3] = \psi(\varphi(x))[4\mathbf{s} + 3]$.

Λ: Find the least k such that $\Lambda^{L(A)}(k)[4\mathbf{s}+3]$ is not defined yet. If k is a stage at which no number is enumerated into D, then define $\Lambda^{L(A)}(k) = 0$, with use $\lambda(k) = \lambda(k-1)[4\mathbf{s}+3]$. If a number n is enumerated into D at stage k, then define $\Lambda^{L(A)}(k) = L(D)(k)[4\mathbf{s}+3]$, with use $\lambda(k) = 4\mathbf{s}+3$, if $\Lambda^{L(A)}(k)$ has not been defined before, or with use $\lambda(k) = s'$, where s' is the last working stage at which some number $x' < \psi(\varphi(n))[s'']$ leaves A, where s'' is the previous use of $\lambda(k)$.

Θ: Find the least k such that $\Theta^{L(B)}(k)[4\mathbf{s}+3]$ is not defined yet. If k is a stage at which no number is enumerated into D, then define $\Theta^{L(B)}(k) = 0$, with use $\theta(k) = \theta(k-1)[4\mathbf{s}+3]$. If a number n is enumerated into D at stage k, then define $\Theta^{L(B)}(k) = L(D)(k)[4\mathbf{s} + 3]$, with use $\theta(k) = 4\mathbf{s} + 3$, if $\Theta^{L(B)}(k)$ has not been defined before, or with use $\theta(k) = s'$, where s' is the last working stage at which some number $y' < \varphi(n)[s'']$ leaves B, where s'' is the previous use of $\theta(k)$.

This completes the work of block $[4\mathbf{s}, 4\mathbf{s} + 3]$. Go to the next block.

4 Verification

We now prove that the constructed set D, and that all the p.c. functionals $\Gamma, \Delta, \Lambda, \Theta$, satisfy all the requirements. Obviously, α is total.

Lemma 1. *Γ^D is well-defined and total, and $\Gamma^D = A$.*

Proof. Fix x. By the construction, $\gamma(x)$ is kept as $2x+1$, and hence $\Gamma^D(x)$ is undefined, only when some $x' \leq x$ enters A or leaves A, which can happen only finitely many times. Also we enumerate $2x$ or $2x+1$ into D, only when (or after) x enter A or leaves A. When $2x$ (maybe together with $2x+1$) is removed from D, x already left A, and removing $2x$ from D recovers $\Gamma^D(x)$ to a previous computation, i.e. a computation with value 0, and hence we have $\Gamma^D(x) = A(x)$. This shows that $\Gamma^D(x)$ is defined and equals to $A(x)$.

This proves that $\Gamma^D = A$.

Lemma 2. *Δ^A is well-defined and total, and $\Delta^A = D$.*

Proof. Fix x. By construction, $\delta(2x)$ and $\delta(2x+1)$ depend on the use $\psi(\varphi(x))$. By our assumption on $A = \Phi^B$ and $B = \Psi^A$, Φ^B and Ψ^A are both total, and hence $\varphi(x)$ and $\psi(u)$ for all $u \leq \varphi(x)$ settle down after a big stage. As we only define $\Delta^A(2x)$ and $\Delta^A(2x+1)$ at a working stage t, when they have no definition at this stage. It can happen that $\Delta^A(2x)[t']$ or $\Delta^A(2x+1)[t']$, with $t' < t$, become valid, after stage t above, at a working stage t'' say, i.e.

$$A_{t''}\lceil \delta(2x)[t'] = A_{t'}\lceil \delta(2x)[t'],$$

then $\Delta^A(2x)[t'']$ and $\Delta^A(2x)[t']$ are actually the same, which means some number $m < \delta(2x)[t']$ leaves A between stage t and stage t'', and hence $\Delta^A(2x)[t]$ become invalid automatically, as A and $A[t]$ can never agree on the initial segment $A_t\lceil \delta(2x)[t']$ after stage t''. This shows that $\Delta^A(2x)$ and $\Delta^A(2x+1)$ are both well-defined.

We now prove that $\Delta^A(2x) = D(2x)$.

If x never enters A, then $2x$ never enters D, which means that whenever $\Delta^A(2x)$ is defined, it is defined as 0. So we assume that x enters A, at a working stage $4\mathbf{s}$ say. Then $\Delta^A(2x)$ is undefined, because of this enumeration. If at this stage, $2x$ is enumerated into D immediately, then we can redefine $\Delta^A(2x)$ as 1. It can also happen that at this stage, a number n (even, or odd) is enumerated into D, and this n is a support of x. Note that such a support can have changes, due to the changes of A, and it could be possible that before x becomes a support of itself, x already leaves A, and if so, whenever we define $\Delta^A(2x)$, we just defined as 0. If x becomes a support of itself, eventually, at a working stage $\mathbf{s}'$ say, then at this stage, $\Delta^A(2x)$ should have no definition, and we can enumerate $2x$ into D. It is actually true, because when x enters A, $\Delta^A(2x)$ is undefined by $x < \delta(2x)$, and then at any stage, when x has a new support, the previous support of x leaves and the new support is still less than x, and these changes, as they are all less than x, actually undefine $\Delta^A(2x)$. Thus, at stage $\mathbf{s}'$, $\Delta^A(2x)$ is not defined, and we are allowed to enumerate $2x$ into D and redefine $\Delta^A(2x)$ as 1.

Now, again because of A-changes, at a working stage $\mathbf{s}'' > \mathbf{s}'$, $\Delta^A(2x)$ could recover to a computation before stage $\mathbf{s}'$, and hence $\Delta^A(2x)[\mathbf{s}''] = 0$. If it happens, then, by construction, $2x$ will be removed out of D at stage $\mathbf{s}''$. We note that at stage $\mathbf{s}''$, $\Delta^A(2x)$ is actually recovered to a computation before $4\mathbf{s}$, because at any stage between $4\mathbf{s}$ and $\mathbf{s}'$, some number less than x, i.e., x's support is in A, and by stage $\mathbf{s}'$, all these numbers are not in A anymore. This means that at stage $\mathbf{s}''$, x is not in A neither, and hence the removal of $2x$ from D is consistent with the definition of Δ^A.

We comment here that after stage $\mathbf{s}''$, $\Delta^A(2x)$ cannot come back to a computation between $4\mathbf{s}$ and $\mathbf{s}''$, because x leaves A, and also some other numbers (at stage $\mathbf{s}''$, all these numbers should already leave A), and as A is d.c.e., these numbers will leave A forever, and hence ensure that $\Delta^A(2x)$ cannot come back to a computation between $4\mathbf{s}$ and $\mathbf{s}''$, which, if true, could require to enumerate $2x$ into D again.

Now we consider $\Delta^A(2x+1) = D(2x+1)$. Without loss of generality, we assume that $2x+1$ is enumerated into D at a working stage $\mathbf{s}'$. Then by construction, we know that by stage $\mathbf{s}'$, x is already a support of itself, and x leaves A by this stage. Because $2x$ is not removed from A, we know $\Delta^A(2x)$ does not recover to any computation before x enters A. That is, before x leaves A, some number z less than $\psi(\varphi(x))$ enters or leaves A. This change allows us to enumerate $2x+1$ into D, which is consistent with the definition of Δ^A.

If z leaves A, then $\Delta^A(2x)$ can never recover to any computation of it before x enters A, and $2x$, $2x+1$ will remain in D forever. That is, after stage $\mathbf{s}'$, whenever we define $\Delta^A(2x)$, $\Delta^A(2x+1)$ again, we just define it as $D(2x), D(2x+1)$, we just let

$$\Delta^A(2x) = D(2x), \quad \Delta^A(2x+1) = D(2x+1),$$

with use $\psi(\varphi(x))$. This ensures $\Delta^A(2x+1) = D(2x+1)$.

If z enters A, and stays in A, then just as discussed above, we can define $\Delta^A(2x) = D(2x)$, $\Delta^A(2x+1) = D(2x+1)$, with use $\psi(\varphi(x))$. This ensures $\Delta^A(2x+1) = D(2x+1)$.

So we assume that z enters A first and later leaves A, and also that at stage $\mathbf{s}''$, $\Delta^A(2x)$ recovers to a previous computation, before x enters A. Then, by construction, $2x, 2x+1$ are removed from D. We have seen why $2x$ can be removed, and for $2x+1$, we know that at stage $\mathbf{s}'$, z enters A, allowing us to enumerate $2x+1$ into D, and when we define $\Delta^A(2x+1)$ again, $\delta(2x+1)$ is bigger than this z. Thus, when z leaves A, $\Delta^A(2x+1)$ is undefined, and especially, at stage $\mathbf{s}''$, $\Delta^A(2x+1)$ is not defined as 1, and hence we can remove $2x+1$ out of D. Again, as discussed for $\Delta^A(2x)$, $\Delta^A(2x+1)$ will not recover to a computation between $\mathbf{s}'$ and $\mathbf{s}''$, and will not require $D(2x+1) = 1$ in the remainder of the construction.

This completes the proof that Δ^D is well-defined and computes A correctly.

Lemma 3. *$\Lambda^{L(A)}$ is well-defined and $L(D) = \Lambda^{L(A)}$.*

Proof. Fix s. Here we consider that s is a general stage, not necessary to be working stage.

If at stage s, no number is enumerated into D, then we always define $\Lambda^{L(A)}(s)$ as 0, with use $\lambda(s)$ the same as $\lambda(s-1)$. Thus, if $\Lambda^{L(A)}(s-1)$ is defined, then so is $\Lambda^{L(A)}(s)$, with value 0. This gives that $\Lambda^{L(A)}(s) = L(D)(s)$, as s can never enter $L(D)$.

So we assume that a number $2x$ is enumerated into D at stage s, i.e. $s^D(2x) = s$. The case that an odd number enters D at stage s can be argued in a similar way, and we leave it to the reader to verify this.

Note that $2x$ is put into D at stage s, x is already in A, and assume that x enters A at stage $s^A(x)$.

We first consider the case when x enters A at stage s. As we also assume that $2x$ also enters D at stage s, x is the support of itself. When we define $\Lambda^{L(A)}(s)$, we define it as 0 with use $\lambda(s) = s$. This use will be kept the same, till s enters $L(A)$ (i.e. x leaves A) at stage s' say. (So, if x remains in A, then such a stage s' does not exist, $\Lambda^{L(A)}(s)$ will have value 0, which equals to $L(D)(s)$.) Then, at stage s', $\Lambda^{L(A)}(s)$ is undefined, and also at (perhaps before) stage s', we do have numbers $x' < \psi(\varphi(x))[s]$ entering A (if some number enters A before stage s and leaves A at stage s', $2x$ will be kept in D forever, by the construction), and when we redefine $\Lambda^{L(A)}(s)$, we define it as 0, with use $\lambda(s) = s'$. Again, the construction ensures that $2x$ will be in D, until x' leaves A, at stage s'', which means that $s^A(x')$ enters $L(A)$ at stage s'', undefining $\Lambda^{L(A)}(s)$. Of course, if $2x$ is not removed from D at stage s'', then another number $x'' < \psi(\varphi(x))[s]$ enters A by stage s'' and thus, when we redefine $\Lambda^{L(A)}(s)$, we define it as 0, with use $\lambda(s) = s''$. As $\psi(\varphi(x))[s]$ is fixed, such a process can be repeated at most finitely many times, and will stop after a certain stage big enough, s^*. Thus, by stage s^*, either s is already in $L(D)$, or s will never be enumerated into $L(D)$. This again gives that $\Lambda^{L(A)}(s)$ is defined and equals to $L(D)(s)$.

Now consider the case that $s^A(x) < s$. Then after stage s, when we check the change below use, the use is $\psi(\varphi(x))[s^A(x)]$, as those computations between stage $s^A(x)$ and stage s can never be recovered after stage s. An almost same argument shows that $\Lambda^{L(A)}(s)$ is defined and equals to $L(D)(s)$.

Lemma 4. *$\Theta^{L(B)}$ is well-defined and $L(D) = \Theta^{L(B)}$.*

Proof. The basic idea of the proof is similar to that in the proof of Lemma 3, but here we consider the changes of B below the use $\varphi(x)$.

This completes the proof of Theorem 3.

References

1. Fang, C.L., Wu, G., Yamaleev, M.M.: On a problem of Ishmuhkametov. Arch. Math. Logic **52**, 733–741 (2013)
2. Ishmukhametov, S.: On the predececcors of d.r.e. degrees. Arch. Math. Logic **38**, 373–386 (1999)
3. Ishmukhametov, S.: On relative enumerability of turing degrees. Arch. Math. Logic **39**, 145–154 (2000)
4. Soare, R.: Recursively Enumerable Sets and Degrees. Springer, Berlin (1987)