Definable Filters in the Structure of Bounded Turing Reductions

Angsheng Li[1,⋆], Weilin Li[1,2,⋆], Yicheng Pan[1,2,⋆], and Linqing Tang[1,2,⋆]

[1] State Key Lab. of Computer Science, Institute of Software, Chinese Academy of Sciences
[2] Graduate University of Chinese Academy of Sciences
{angsheng,weilin,yicheng,linqing}@ios.ac.cn

Abstract. In this article, we show that there exist c.e. bounded Turing degrees $\mathbf{a}$, $\mathbf{b}$ such that $\mathbf{0} < \mathbf{a} < \mathbf{0}'$, and that for any c.e. bounded Turing degree $\mathbf{x}$, $\mathbf{b} \vee \mathbf{x} = \mathbf{0}'$ if and only if $\mathbf{x} \geq \mathbf{a}$. The result gives an unexpected definability theorem in the structure of bounded Turing reducibilities.

1 Introduction

A set $A \subseteq \omega$ is called *computably enumerable* (c.e., for short), if there is an algorithm to enumerate the elements of it. Given sets $A, B \subseteq \omega$, we say that A is Turing *reducible to* B, if there is an oracle Turing machine, Φ say, such that $A = \Phi^B$ (denoted by $A \leq_{\mathrm{T}} B$), and furthermore, if the bits of oracle queries are bounded by a computable function, we say that A is *bounded Turing reducible* to B (written $A \leq_{\mathrm{bT}} B$). A Turing and a bounded Turing (or bT, for short) degree is the equivalence class of a set under the Turing reductions and the bounded Turing reductions respectively. A degree is called *computably enumerable* (c.e.), if it contains a c.e. set.

Let $\mathcal{E}$ and $\mathcal{E}_{\mathrm{bT}}$ be the structures of the c.e. degrees under the Turing reductions and the bounded Turing reductions respectively. During the past decades, the studies of both structures focused on that of the algebraic properties, leading to major achievements such as the decidability results of the Σ_1-theory of $\mathcal{E}$, and the Σ_2-theory of $\mathcal{E}_{\mathrm{bT}}$ (Ambos-Spies, P. Fejer, S. Lempp and M. Lerman [1996]), and the undecidability results of the Σ_3-theory of $\mathcal{E}$ (Lempp, Nies, and Slaman [1998]), and of the Σ_4-theory of $\mathcal{E}_{\mathrm{bT}}$ (Nies and Lempp [1995]). This progress brings the decidability problems of the Σ_2-theory of $\mathcal{E}$, and the Σ_3-theory of $\mathcal{E}_{\mathrm{bT}}$ into sharper focus, for which new ingredients are welcome.

In recent years, the study of the computably enumerable degrees has focused on Turing definability in the structure $\mathcal{E}$. For instance, Slaman asked in 1985 if there are any c.e. degrees that are incomplete and nonzero which are definable in the c.e. degrees $\mathcal{E}$. A natural approach to this problem is to find some definable substructures of $\mathcal{E}$ that have nontrivial minimal/maximal and/or least/greatest

⋆ The authors are partially supported by NSFC Grant No. 60325206, and No. 60310213.

M. Agrawal et al. (Eds.): TAMC 2008, LNCS 4978, pp. 116–124, 2008.

members. This resumes interests in topics such as the continuity of the c.e. degrees, started by Lachlan early in 1967.

Harrington and Soare [1992] showed that there is no maximal minimal pair in the c.e. Turing degrees, and Seetapun [1991] proved a stronger result that for any c.e. degree $\mathbf{b} \neq \mathbf{0}, \mathbf{0}'$, there is a c.e. Turing degree $\mathbf{a} > \mathbf{b}$ such that for any c.e. Turing degree $\mathbf{x}$, $\mathbf{a} \wedge \mathbf{x} = \mathbf{0}$ if and only if $\mathbf{b} \wedge \mathbf{x} = \mathbf{0}$.

In the dual case, Ambos-Spies, Lachlan and Soare [1993] showed that for any c.e. Turing degrees $\mathbf{x}, \mathbf{y}$, if $\mathbf{x}$ and $\mathbf{y}$ are nontrivial splitting of $\mathbf{0}'$, then there exists a c.e. Turing degree $\mathbf{a} < \mathbf{x}$ such that $\mathbf{a} \vee \mathbf{y} = \mathbf{0}'$. Remarkably, Cooper and Li [ta] (the major subdegree theorem) showed that for any c.e. Turing degree $\mathbf{b} \neq \mathbf{0}, \mathbf{0}'$, there exists a c.e. Turing degree $\mathbf{a} < \mathbf{b}$ such that for any c.e. Turing degree $\mathbf{x}$, $\mathbf{b} \vee \mathbf{x} = \mathbf{0}'$ if and only if $\mathbf{a} \vee \mathbf{x} = \mathbf{0}'$ (.

However, the study of the continuity and the definability in the bounded Turing structures is started just recently. Paul Brodhead, Angsheng Li and Weilin Li [ta] (BLL) showed that the Seetapun's result holds in the c.e. bounded Turing degrees, that is, for any c.e. bounded Turing (bT) degree $\mathbf{b} \neq \mathbf{0}, \mathbf{0}'$, there exists a c.e. bT-degree $\mathbf{a} > \mathbf{b}$ such that for any c.e. bT-degree $\mathbf{x}$, $\mathbf{a} \wedge \mathbf{x} = \mathbf{0}$ if and only if $\mathbf{b} \wedge \mathbf{x} = \mathbf{0}$. In the present paper, we consider the dual case of this result, the BLL result. It is very unexpected that the dual case fails badly. In fact we are able to prove:

Theorem 1. *There exist c.e. bounded Turing degrees $\boldsymbol{a}, \boldsymbol{b}$ with the following properties:*

(1) $\boldsymbol{0} < \boldsymbol{a} < \boldsymbol{0}'$,

(2) For any c.e. bounded Turing degree $\boldsymbol{x}$, $\boldsymbol{b} \vee \boldsymbol{x} = \boldsymbol{0}'$ if and only if $\boldsymbol{x} \geq \boldsymbol{a}$.

An immediate corollary of the theorem is that there exist c.e. bounded Turing degrees $\mathbf{a}$, $\mathbf{b}$ such that $\mathbf{a} \vee \mathbf{b} = \mathbf{0}'$ but for no c.e. bT-degree $\mathbf{x} < \mathbf{a}$ with $\mathbf{x} \vee \mathbf{b} = \mathbf{0}'$, and that the Cooper-Li major subdegree theorem fails badly in the c.e. bT-degrees.

However the result gives a very nice theorem in the Turing definability of the c.e. bounded Turing degrees. That is, there is a principal filter $[\mathbf{a}, \mathbf{0}']$ which is definable by equation $\mathbf{x} \vee \mathbf{b} = \mathbf{0}'$ for some nonzero and incomplete c.e. bT-degree $\mathbf{b}$. The result may provide ingredients to the decidability/undecidability problem of the Σ_3-theory of the c.e. bounded Turing degrees.

The rest of this paper is devoted to proving theorem 1.1, our main result. In section 2, we formulate the conditions of the theorem by requirements, and for each requirement, we give corresponding strategy to satisfy it; in section 3, we arrange all strategies to satisfy the requirements on nodes of a tree, or more precisely, the *priority tree* T. In section 4, we use the priority tree to describe a stage-by-stage construction of the objects we need. Finally, in section 5 we verify that the construction in section 4 satisfies all of the requirements, finishing the proof of the theorem.

Our notation and terminology are standard and generally follow Soare [1987]. During the course of a construction, notations such as A, Φ are used to denote the current approximations to these objects, and if we want to specify the values

immediately at the end of stage s, then we denote them by A_s, $\Phi[s]$ etc. For a *partial computable* (p.c., or for simplicity, also a Turing) functional, Φ say, the use function is denoted by the corresponding lower case letter ϕ. The value of the use function of a converging computation is the greatest number which is actually used in the computation. For a Turing functional, if a computation is not defined, then we define its use function $= -1$. During the course of a construction, whenever we define a parameter, p say, as *fresh*, we mean that p is defined to be the least natural number which is greater than any number mentioned so far. In particular, if p is defined as fresh at stage s, then $p > s$. The notion of bounded Turing reducibility is taken from Soare's new book: Computability Theory and Applications [ta] (Definition 3.4.1).

2 Requirements and Strategies: Theorem 1.1

2.1 The Requirements

We will build c.e. sets A, B, C and D to satisfy the following requirements:

$$\begin{array}{ll} \mathcal{T}: & K \leq_{\mathrm{bT}} A \oplus B \\ \mathcal{P}_e: & A \neq \theta_e \\ \mathcal{S}_e: & C \neq \Psi_e(A) \\ \mathcal{R}_e: & D = \Phi_e(X_e, B) \longrightarrow\ A \leq_{\mathrm{bT}} X_e \end{array}$$

where $e \in \omega$, $\{(\theta_e, \Phi_e, \Psi_e, X_e) : e \in \omega\}$ is an effective enumeration of all quadruples (θ, Φ, Ψ, X) of all computable partial functions θ, of all bounded Turing (bT, for short) reductions Φ, and Ψ, and of all c.e. sets X; and K is a fixed creative set.

Let $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{x}$ be the bT-degrees of A, B and X respectively. By the $\mathcal{P}$-requirements and the $\mathcal{S}$-requirements, we have $\mathbf{0} < \mathbf{a} < \mathbf{0}'$. By the $\mathcal{R}$-requirements, we know that for any c.e. bT degree $\mathbf{x}$, if $\mathbf{x} \vee \mathbf{b} = \mathbf{0}'$, then $\mathbf{a} \leq \mathbf{x}$. By the $\mathcal{T}$-requirement, for any c.e. bounded Turing degree $\mathbf{x}$, if $\mathbf{a} \leq \mathbf{x}$, then $\mathbf{0}' \leq \mathbf{a} \vee \mathbf{b} \leq \mathbf{x} \vee \mathbf{b}$. Therefore satisfying all the requirements is sufficient to prove the theorem.

We introduce some conventions of the bounded Turing reductions for describing the strategies. We will assume that for any given bounded Turing reduction Φ or Ψ, the use functions ϕ and ψ will be increasing in arguments. Now we are ready to describe the strategies.

2.2 The $\mathcal{T}$-Strategy

To satisfy the $\mathcal{T}$-requirement, $K \leq_{\mathrm{bT}} A \oplus B$, we need to construct a bounded Turing reduction Γ such that $K = \Gamma(A, B)$.

We construct the Γ by coding K into A and B as follows:

For any k, if $k \in K$, then either $2k \in A$ or $2k \in B$.

Therefore, $k \in K$ if and only if either $A(2k) = 1$ or $B(2k) = 1$. K is bounded Turing reducible to $A \oplus B$, the $\mathcal{T}$-requirement is satisfied.

2.3 A $\mathcal{P}$-Strategy

A $\mathcal{P}$-strategy satisfying a $\mathcal{P}$-requirement, $A \neq \theta$ say, is a Friedberg-Muchnik procedure, and proceeds as follows:

1. Define a witness a as a fresh odd number.
2. Wait for a stage, v say, at which $\theta(a) \downarrow = 0 = A(a)$, then
   — enumerate a into A and terminate.

By the strategy, we know if step 2 of the procedure occurs, then $\theta(a) \downarrow = 0 \neq 1 = A(a)$ is created, otherwise $\theta(a) \neq 0 = A(a)$ occurs at each stage. In either case, the $\mathcal{P}$-requirement is satisfied. We use a node on a tree (the priority tree as we'll see later), γ say, to denote a $\mathcal{P}$-strategy.

2.4 An $\mathcal{S}$-Strategy

Suppose that we want to satisfy an $\mathcal{S}$-requirement, $C \neq \Psi(A)$ say. It will be satisfied by a Friedberg-Muchnik procedure as follows:

1. Define a witness c as a fresh odd number.
2. Wait for a stage, v say, at which $\Psi(A; c)[v] \downarrow = 0 = C_v(c)$, then
   — enumerate c into C;
   — define an A-restraint $r^A = \psi(c)$.

The key point to the satisfaction of $\mathcal{S}$ is that the A-restraint $r^A = \psi(c)$ will be preserved once step 2 of the strategy occurs. In this case, we have that if step 2 occurs, then $\Psi(A; c) \downarrow = 0 \neq 1 = C(c)$ is created and preserved, else if step 2 never occurs, then $\Psi(A; c) \neq 0 = C(c)$. In either case, the $\mathcal{S}$-requirement is satisfied. We use a node β, say, on the priority tree to denote the $\mathcal{S}$-strategy.

2.5 An $\mathcal{R}$-Strategy

Suppose we want to satisfy an $\mathcal{R}$ requirement, $\mathcal{R}$: $D = \Phi(X, B) \longrightarrow A \leq_{\mathrm{bT}} X$, say. We will build a bounded Turing reduction Δ such that if $D = \Phi(X, B)$ then $\Delta(X) = A$. The bounded Turing reduction Δ will be built by an ω-sequence of cycles n, cycle n will be responsible for defining $\Delta(X; n)$.

The $\mathcal{R}$-strategy will proceed as follows:

1. Let n be the least x, such that $\Delta(X; x) \uparrow$. Define a *witness block* U_n such that $min\{y \in U_n\}$ is fresh, and that $|U_n| = n + 1$.
2. Wait for a stage, v say, at which the following conditions occur: For all $y \in U_{n'}$ for some $n' \leq n$, we have
   $$\Phi(X, B; y) \downarrow = D(y)$$
   then:
   — define $\Delta(X; n) \downarrow = A(n)$ with $\delta(n) = max\{\phi(y) | y \in U_{n'}, n' \leq n\}$.
3. If there is an x such that $\Delta(X; x) \downarrow \neq A(x)$, then let n be the least such x, and go on to step 4.

4. If for all $y \in U_n$, $\Phi(X, B; y) \downarrow = D(y)$, then
   — let x be the least $y \in U_n \backslash D$;
   — enumerate x into D;
   — define a *conditional restraint* $\overrightarrow{r} = (n, \delta(n)]$.
5. Suppose that a conditional restraint $\overrightarrow{r} = (n, \delta(n)]$ is kept, then there is no element b with $n < b \leq \delta(n)$ that can be enumerated into B.

Note that if X changes below $\delta(n)$, then $\Delta(X; n)$ becomes undefined, and simultaneously the conditional restraint $\overrightarrow{r} = (n, \delta(n)]$ drops.

The new idea of this proof is the notion of the witness block, which seems the first time it becomes available, and the notion of conditional restraints. The later notion was largely the first author's idea that has already been used several times in the literature.

Now we analyze the correctness of the $\mathcal{R}$-strategy. We proceed the arguments by cases:

Case 1: $\Delta(X)$ is built infinitely often.

In this case we will prove that $\Delta(X)$ is a total function (i.e., $\Delta(X; x)$ is defined for every x). Note that for every n, $\Delta(X; n)$ is redefined only finitely many times. So there exists some s , such that for any stage s' with $s' > s$, $\Delta(X; n)[s']$ does not change any more.

Assume by contradiction, there exists an x such that $\Delta(X; x) \uparrow$ eventually, let n be the least such x. Assume after stage v, the $\Delta(X; n)$ would never been defined any more. By step 2 of the $\mathcal{R}$-strategy, we know that for each stage s with $s \geq v$, there exists some $y \in U_{n'}$ for some $n' \leq n$ such that $\Phi(X, B; y) \downarrow \neq D(y)$ (otherwise $\Delta(X; n)$ will be defined again by step 2 of the $\mathcal{R}$-strategy), then for any undefined $\Delta(X; m)$ with $m > n$, it would never be defined after stage s with $s \geq v$. So $\Delta(X)$ is finitely built, a contradiction. That means that $\Delta(X)$ is a total function.

Now if $\Delta(X) \neq A$, i.e., there exists an x such that $\Delta(X; x) \downarrow \neq A(x)$, we prove that $\Phi(X, B) \neq D$ under this assumption.

Let n be the least such x, we prove that there is an $x \in U_n$ such that $\Phi(X, B; x) \downarrow = 0 \neq 1 = D(x)$. Let s_n be the stage at which the permanent computation $\Delta(X; n)$ was created, and let $t_n > s_n$ be the stage at which n is enumerated into A. By step 4 of the $\mathcal{R}$-strategy, for any $s > t_n$, if $\Phi(X, B; y) \downarrow = D(y)$ holds for all $y \in U_n$, then we enumerate an element $x \in U_n$ into D, and create a conditional restraint $(n, \delta(n)]$. By the choice of s_n, the conditional restraint will be kept forever. By the conditional restraint, no number x with $n < x \leq \delta(n)$ could be enumerated into B, therefore $r_n \doteq \{\Phi(X, B; y) | y \in U_n\}$ can be injured only by numbers $x \leq n$. Once r_n is injured, we may waste a witness $x \in U_n$. However, r_n can be injured at most n many times and $|U_n| = n + 1$. Therefore after up to n many times injury, there is a witness $y \in U_n$ which can be used to create a permanent inequality between $\Phi(X, B)$ and D. Let $v_n > t_n$ be a minimal stage after which B will never change below n. Therefore if at a stage $s > t_n$ we have that

(1) $\Delta(X;n)\downarrow \neq A(n)$, and
(2) for any $y \in U_n$, $\Phi(X,B;y)[s]\downarrow = D_s(y)$,

then we enumerate the least $x \in U_n \backslash D$, x_0 say, into D. By the choice of n, v_n, and by the conditional restraint $\overrightarrow{r} = (n, \delta(n)]$, $\Phi(X,B;x_0)\downarrow = 0 \neq 1 = D(x_0)$ holds permanently, which means $\Phi(X,B)\downarrow \neq D$.

Case 2: Δ is finitely built.

In this case, assume w.l.o.g. that $\Delta(X;0)\downarrow, \cdots, \Delta(X;n-1)\downarrow$, and $\Delta(X;n)\uparrow$ eventually, we can assume after s_n, $\Delta(X;n)$ doesn't have chance to be defined any more (since if it has infinitely many chances to be defined, then it must be defined). By step 2 of the $\mathcal{R}$-strategy, for any stage s with $s > s_n$, there exists a $y \in U_{n'}$ for some $n' \leq n$ such that $\Phi(X,B;y) \neq D(y)$ (otherwise $\Delta(X;n)$ will be defined again by step 2 of the $\mathcal{R}$-strategy). Since $\bigcup_{n' \leq n} U_{n'}$ is finite, so there exists some n' with some $y_0 \in U_{n'}$ such that $\Phi(X,B;y_0)[s] \neq D_s(y_0)$ for infinitely many stages s, that means $\Phi(X,B) \neq D$.

In either case, the $\mathcal{R}$-requirement is satisfied. We use a node on the priority tree, α say, to denote the $\mathcal{R}$-strategy.

Note in the case $\Delta(X) = A$, there is no permanent conditional restraint $\overrightarrow{r}$. However, there may be infinitely many stages at which we create $\overrightarrow{r[s]} = (a[s], b[s]]$. The key to the proof is that this unbounded conditional restraints $\overrightarrow{r[s]}$ are harmless, because:

(1) A conditional restraint $\overrightarrow{r[s]} = (a[s], b[s]]$ controls elements x only if $a[s] < x \leq b[s]$.

(2) The conditional restraints ensure that the $\mathcal{R}$-strategy has no influence on any number no larger than $a[s]$. We say it's not important because in this case, $a[s]$ will be unbounded. This guarantees that the $\mathcal{R}$-strategy would not injure a lower priority $\mathcal{S}$-strategy after some fixed stage. That is to say, a lower priority $\mathcal{S}$-strategy is injured by the $\mathcal{R}$-strategy only finitely many times. This is the reason why we can only use conditional restraints here, instead of the typical restraints, which is of course one of the main contributions of this paper.

We thus define the *possible outcomes of* the $\mathcal{R}$-strategy α by

$$0 <_{\mathrm{L}} 1$$

to denote infinite and finite actions respectively.

3 The Priority Tree

In this section, we will build a *priority tree of strategies* $T \subset \Lambda^{<\omega}$, with $\Lambda = \{0,1\}$. Let $\mathcal{P} < \mathcal{R}$ denote that the priority ranking of $\mathcal{P}$ is higher than that of $\mathcal{R}$. Also let $<_L$ be a *left-to-right* ordering of the nodes on the priority tree, as given below.

Definition 1. Define the *priority ranking of the requirements* such that $\mathcal{T}$ has the highest priority, and $\forall e \in \omega$: $\mathcal{R}_e < \mathcal{S}_e < \mathcal{P}_e < \mathcal{R}_{e+1}$.

Note that both a $\mathcal{P}$-strategy and an $\mathcal{S}$-strategy have only one possible outcome, we denote it by 1. However, as we mentioned before, an $\mathcal{R}$-strategy has two outcomes which we denote by $0 <_L 1$.

We now define the priority tree T inductively as follows.

Definition 2 (The Priority Tree).

(i) Define the root node $\emptyset$ of the tree to be an $\mathcal{R}_0$-strategy.
(ii) The immediate successors of a node are the possible outcomes of the corresponding strategy. We say a requirement $\mathcal{X}$ is satisfied at some node ξ, if there is a node $\xi' \subseteq \xi$ working on the requirement $\mathcal{X}$.
(iii) The immediate successors of a node, ξ say, will work on the highest priority ranking requirement which is not satisfied on path ξ.

Definition 3. The *index $I(\xi)$ of a node ξ* is the index of the requirement on which the node acts. For example, if ξ is an $\mathcal{R}_e$- or a $\mathcal{P}_e$- or an $\mathcal{S}_e$-strategy, we define $I(\xi) = e$.

4 The Construction

Our construction will perform different actions at even and odd stages. Suppose that K is enumerated at odd stages only, and that there is exactly one element that enters K at each odd stage. At even stages, strategies on the tree will act to satisfy the requirements.

During the course of the construction, we may initialize a node, ξ say, which means that all the actions taken by ξ previously, are canceled, or set to be totally undefined.

Now we give the construction stage-by-stage.

Definition 4. (The Construction) The construction is defined as follows:

Stage $s = 0$. Set $A = B = C = D = \emptyset$, and initialize every node of the priority tree T.

Stage $s = 2n + 1$. For $k_s \in K_s \backslash K_{s-1}$. Let $x = 2k_s$, we need to decide either $x \in A$ or $x \in B$:

Case 1: Find the $<$-least ξ such that (a) or (b) below occurs:

(a) $\xi = \alpha$ is an $\mathcal{R}$-strategy such that α has conditional restraint $\overrightarrow{r[s]} \downarrow = (a[s], b[s]]$ and $a[s] < x \leq b[s]$;

(b) $\xi = \beta$ is an $\mathcal{S}$-strategy such that $r^A(\beta) \downarrow \geq x$.

then:

Subcase 1: $\xi = \alpha$, i.e., (a) occurs

— enumerate x into A;

— initialize all nodes ξ to the right of $\alpha\hat{}\langle 0 \rangle$.

Subcase 2: $\xi = \beta$, i.e., (b) occurs

— enumerate x into B;
— initialize all nodes ξ with $\xi \not\leq \beta$.

Case 2: Otherwise, we can't find node ξ such that (a) or (b) occurs for ξ:
— enumerate x into A.

Stage $s = 2n + 2$. We specify certain strategies to be eligible to act. First we allow the root node to be eligible to act at substage $t = 0$. At each substage t, we let the strategy be eligible to act run its program, and then, either close the current stage or specify a node to be eligible to act next, i.e., at the next substage of stage s.

Substage t. Suppose that node ξ is eligible to act at substage t of stage s. If $t = s$, then initialize all nodes $\xi' \not\leq \xi$, and close the current stage. Otherwise, corresponding to different types of the strategy, there are three cases:

Case 1. $\xi = \beta$ is an $\mathcal{S}$-strategy. Then run the following:

Program β:

1. If $c(\beta)\downarrow$, and $\Psi_\beta(A; c(\beta))\downarrow = 0 = C(c(\beta))$, then
   (a) If for any α with $\alpha^\frown\langle 0\rangle \subseteq \beta$ we have $dom(\alpha) > \psi_\beta(c)$, then
   — enumerate $c(\beta)$ into C, and
   — define $r^A(\beta) = \psi(c(\beta))$,
   where $dom(\alpha)$ is the size of the domain of Δ_α.
   (b) Otherwise, then do nothing,
   in either case, initialize all $\xi' \not\leq \beta$ and go to stage $s + 1$.
2. If $c(\beta)\uparrow$, then
   — define $c(\beta)$ as fresh;
   — initialize all $\xi' \not\leq \beta$ and go to stage $s + 1$.
3. Otherwise, let $\beta^\frown\langle 1\rangle$ be eligible to act next.

Case 2. $\xi = \gamma$ is a $\mathcal{P}$-strategy. Run the following

Program γ:

1. If $a(\gamma)\downarrow$, and $\theta_\gamma(a(\gamma))\downarrow = 0 = A(a(\gamma))$, then
   — enumerate $a(\gamma)$ into A,
   – initialize all nodes ξ with $\xi \not\leq \gamma$, and go to stage $s + 1$.
2. If $a(\gamma)\uparrow$, then
   — define $a(\gamma)$ as fresh;
   — initialize all $\xi' \not\leq \gamma$ and go to stage $s + 1$.
3. Otherwise, let $\gamma^\frown\langle 1\rangle$ be eligible to act next.

Case 3. $\xi = \alpha$ is an $\mathcal{R}$-strategy. In this case, we perform the following

Program α:

1. If there is a $k > b(\alpha)$, such that $\Delta_\alpha(X_\alpha; k)\downarrow \neq A(k)$, then:
   (a) If $\overrightarrow{r}(\alpha)\downarrow$, then:
   — Let $\alpha^\frown\langle 1\rangle$ be eligible to act next.

(b) Otherwise and $\forall y \in U^\alpha_k$, $\Phi_\alpha(X_\alpha, B; y)[s] \downarrow = D(y)$, then:
— let i be the least k such that $\Delta_\alpha(X_\alpha; k) \downarrow \neq A(k)$,
— let x be the least $y \in U^\alpha_i \backslash D$, enumerate x into D;
— define a conditional restraint $\overrightarrow{r}(\alpha) = (i, \delta(i)]$;
— initialize all nodes ξ to the right of $\alpha^\frown\langle 0\rangle$, and go to stage $s+1$.
(c) Otherwise:
– let $\alpha^\frown\langle 1\rangle$ be eligible to act next.

2. Otherwise and for $k = \min\{x > b(\alpha) | \Delta_\alpha(X_\alpha; x) \uparrow\}$ we have:
(a) $U^\alpha_k \downarrow$;
(b) $\forall y \in U^\alpha_{k'}$ for some $k' \leq k$, $\Phi_\alpha(X_\alpha, B; y)[s] \downarrow = D(y)$.
Then:
— define $\Delta_\alpha(X_\alpha; k) \downarrow = A(k)$ with $\delta_\alpha(k) = \max\{\phi_\alpha(y) | y \in U^\alpha_{k'}, k' \leq k\}$;
— let $\alpha^\frown\langle 0\rangle$ be eligible to act next.
3. Otherwise and for $k = \min\{x > b(\alpha) | \Delta_\alpha(X_\alpha; x) \uparrow\}$ we have $U^\alpha_k \uparrow$, then:
— define $U^\alpha_k = (l, l + k + 1)$, where l is a fresh number;
— initialize all nodes ξ to the right of $\alpha^\frown\langle 0\rangle$, and close the current stage.
4. Otherwise and $b(\alpha) \uparrow$, then:
— define $b(\alpha)$ as fresh;
— initialize all $\xi' \not\leq \alpha$ and go to stage $s+1$.
5. Otherwise, then let $\alpha^\frown\langle 1\rangle$ be eligible to act next.

This completes the description of the construction. The verification of this construction will be given in the full version of the paper.

References

1. Ambos-Spies, K., Fejer, P.A., Lempp, S., Lerman, M.: Decidability of the two-quantifier theory of the recursively enumerable weak truthtable degrees and other distributive upper semi-lattices. Journal of Symbolic Logic 61, 880–905 (1996)
2. Ambos-Spies, K., Lachlan, A.H., Soare, R.I.: The continuity of cupping to 0'. Annals of Pure and Applied Logic 64, 195–209 (1993)
3. Brodhead, P., Li, A., Li, W.: Continuity of capping in ε_{dT} (to appear)
4. Cooper, S.B., Li, A.: On Lachlan's major subdegree problem (to appear)
5. Harrington, L., Soare, R.I.: Games in recursion theory and continuity properties of capping degrees. In: Judah, H., Just, W., Woodin, W.H. (eds.) Set Theory and the Continuum, Proceedings of Workshop on Set Theory and the Continuum, MSRI, Berkeley, CA, October, 1989, pp. 39–62. Springer, Heidelberg (1992)
6. Nies, A., Lempp, S.: The undecidability of the Π_4-theory for the r.e. wtt- and Turing degrees. Journal of Symbolic Logic 60, 1118–1136 (1995)
7. Nies, A., Lempp, S., Slaman, T.A.: The Π_3-theory of the enumerable Turing degrees is undecidable. Transactions of the American Mathematical Society 350, 2719–2736 (1998)
8. Seetapun, D.: Contributions to Recursion Theory, Ph. D thesis, Trinity College (1991)
9. Soare, R.: Recursively Enumerable Sets and Degrees. Springer, Heidelberg (1987)
10. Soare, R.I.: Computability Theory and Applications. Springer, Heidelberg (to appear)