Notre Dame Journal of Formal Logic
Volume 44, Number 1, 2003

Differences between Resource Bounded Degree Structures

Michael E. Mytilinaios and Theodore A. Slaman

Abstract We exhibit a structural difference between the truth-table degrees of the sets which are truth-table above $0'$ and the *PTIME*-Turing degrees of all sets. Though the structures do not have the same isomorphism type, demonstrating this fact relies on developing their common theory.

1 Introduction

For sets A and B, A is recursive in B ($A \leq_T B$) if and only if there exists an algorithm to compute A given complete information about B. If A and B are recursive in each other, we say that they have the same Turing degree. The Turing degree of a set is a measurement of the information which is contained in the diagram of that set. The Turing degrees are partially ordered by $\leq_T$ on their representatives.

By restricting the class of allowed algorithms, we obtain finer notions of degree. For example, A is truth-table reducible to B ($A \leq_{tt} B$) if and only if there is a total recursive function g and an algorithm to compute A from B such that for each n, the algorithm runs in less than $g(n)$ many steps. The truth-table degrees are the associated equivalence classes. Similarly, A is *PTIME*-computable from B if the function g is a polynomial, and the *PTIME*-degrees are the associated equivalence classes. With subexponential time classes, the representation of sets is important; we will always work with sets of finite binary strings and calculate the run time of programs in terms of the lengths of their inputs.

In general, if u is a collection of total recursive functions, we say that $A \leq_u B$ if there is a g in u and an algorithm to determine atomic facts about A from B such that the run time of the algorithm is bounded by g. Typically, u is taken to represent a natural class of time complexity. We let $\mathcal{D}_{tt}$ denote the partial order of the truth-table degrees, $\mathcal{D}_{PTIME}$ the *PTIME*-degrees, and $\mathcal{D}_u$ the u-degrees.

Received December 8, 2002; accepted January 27, 2003; printed April 16, 2004
2000 Mathematics Subject Classification: Primary, 03D28; Secondary, 68Q15
Keywords: Turing degree, polynomial-time degree, truth-table degree

$\mathcal{D}_{tt}$ is an odd member among the above collection of bounded resource degree structures. The other us are uniformly recursive; that is to say, there is a single recursive function $h(n, m)$ of two variables such that for every g, $g \in u$ if and only if there is an m such that for all n, $g(n) = h(n, m)$. However, this difference disappears if we consider $\mathcal{D}_{tt}(\geq_{tt}0')$, the truth-table degrees of sets above the halting problem, and let h be recursive in $0'$.

Downey raised the question whether moving to the degrees above $0'$ removes the differences between $\mathcal{D}_{tt}$ and the more complexity theoretic degree structures $\mathcal{D}_u$.

The answer is both no and yes. We will show that $\mathcal{D}_{tt}(\geq_{tt}0')$ is not isomorphic to $\mathcal{D}_{PTIME}$. However, we come to this conclusion by exploiting the extensive similarities between the two structures.

We will show that $\mathcal{D}_{tt}(\geq_{tt}0')$ and $\mathcal{D}_{PTIME}$ are not isomorphic by showing that $\mathcal{D}_{tt}(\geq_{tt}0')$ is locally more complicated than $\mathcal{D}_{PTIME}$. For this, we will use finite sequences $\boldsymbol{p}$ of degrees to specify infinite sequences. Working in $\mathcal{D}_{tt}(\geq_{tt}0')$, we will show that if $\boldsymbol{p}$ specifies the sequence $\langle g_i : i \in \omega\rangle$, then there is another finite sequence $\boldsymbol{q}$ below the join of $\boldsymbol{p}$ such that $\boldsymbol{q}$ specifies the subsequence $\langle g_i : i \in 0'''\rangle$. However, in $\mathcal{D}_{PTIME}$, there is a $\boldsymbol{p}$ specifying a sequence $\langle g_i : i \in \omega\rangle$ such that for every finite sequence $\boldsymbol{q}$ below the join of $\boldsymbol{p}$, $\boldsymbol{q}$ does not specify the subsequence $\langle g_i : i \in 0'''\rangle$ (in the sense of the previous sentence).

2 Isomorphism Types

2.1 Defining ω-sequences from parameters

Conventions In Section 2.1, we develop a common part of the theories of $\mathcal{D}_{tt}(\geq_{tt}0')$ and $\mathcal{D}_{PTIME}$. The following can be applied equally well in either of the two, so we will refer simply to $\mathcal{D}$. Similarly, we will let O refer to a representative of the least element of $\mathcal{D}$: $\varnothing$ when $\mathcal{D} = \mathcal{D}_{PTIME}$ and $0'$ when $\mathcal{D} = \mathcal{D}_{tt}(\geq_{tt}0')$. Finally, we will use uppercase Greek letters, such as Φ and Ψ, to denote Turing functionals which correspond to reductions of type $\mathcal{D}$ and refer to them as $\mathcal{D}$-functionals. For example, a $\mathcal{D}_{PTIME}$-functional is a Turing functional that runs in polynomial time. To keep the notation uncluttered, we will not explicitly join our sets with O, but we will make the convention that any $\mathcal{D}$-functional can refer to the oracle O.

Definition 2.1

1. An *ideal* in $\mathcal{D}$ is a set $\mathcal{I}$ that is closed under join and closed downward.
2. Intersection gives an operation of meet on ideals. Union followed by closure under $\mathcal{D}$'s join and closure downward gives an operation of join for ideals.
3. Given a k in $\mathcal{D}$, let (k) denote $\{x : x \leq_{\mathcal{D}} k\}$. Clearly, (k) is an ideal. Similarly, if $\mathcal{K}$ is a set of elements in $\mathcal{D}$, let $(\mathcal{K})$ denote the ideal generated by the elements of $\mathcal{K}$.

There are many ways by which finitely many parameters can be used to generate an infinite sequence in $\mathcal{D}$. In Definition 2.2, we specify one such method with features motivated by Shore [6] and Nies et al. [4]. This method is well suited to specifying subsequences from presentation of sequences.

Definition 2.2 A finite sequence $\boldsymbol{p}$ of elements of $\mathcal{D}$ *specifies the infinite sequence* $\langle g_i : i \in \omega\rangle$ if and only if there are sets E_1, F_1, E_2, F_2, D_1, and D_2 which represent the elements of $\boldsymbol{p}$ in order, and there are sets $\langle G_i : i \in \omega\rangle$ which represent the elements of $\langle g_i : i \in \omega\rangle$ in order, and the following conditions hold.

1. For any finite set $G_{n_1}, \dots, G_{n_k}$ and G_m, if for all $j \leq k$, $n_j \neq m$, then $(\{G_{n_1}, \dots, G_{n_k}\}) \cap (G_m) = (O)$.
2. (a) $D_1 \not\geq_{\mathfrak{D}} D_2$, and
   (b) for each $n \in \omega$, $D_1 \oplus G_n \geq_{\mathfrak{D}} D_2$.
3. For each $n \in \omega$,
   (a) if n is odd, then $(F_1 \oplus G_n) \cap (E_1) = (G_{n+1})$, and
   (b) if n is even, then $(F_2 \oplus G_n) \cap (E_2) = (G_{n+1})$.

If $\boldsymbol{p}$ specifies a sequence, then the set of elements of that sequence is not necessarily first-order definable from $\boldsymbol{p}$, but it is associated with $\boldsymbol{p}$ in a way that is invariant under isomorphism.

For the next definition, let $\boldsymbol{p}$ specify the sequence $\langle g_i : i \in \omega \rangle$ and adopt the notation of Definition 2.2. Further, if $\boldsymbol{p} = \langle p_1, \dots, p_k \rangle$ and q is a degree, then let $\boldsymbol{p}^\frown\langle q \rangle$ denote the sequence $\langle p_1, \dots, p_k, q \rangle$ obtained by appending q to $\boldsymbol{p}$.

Definition 2.3 Suppose that q is a degree in $\mathfrak{D}$ and Q is a set of degree q. We say that the sequence $\boldsymbol{p}^\frown\langle q \rangle$ *specifies the subsequence* $\langle g_i : i \in S \rangle$ if and only if, for all i,

$$i \in S \iff (\exists X)[G_i \geq_{\mathfrak{D}} X \text{ and } X \oplus D_1 \geq_{\mathfrak{D}} D_2 \text{ and } Q \geq_{\mathfrak{D}} X].$$

The technology to control meets in $\mathfrak{D}_{PTIME}$ was developed in Ambos-Spies [1]. It was developed further in Shinoda and Slaman [5] and Shore and Slaman [7]. We apply some of that technology in the next theorem. But first we introduce a Skolemized version of Definition 2.2.

Definition 2.4 A *verified sequence* is a finite sequence of $\mathfrak{D}$-Turing functionals $\langle (\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i \rangle$ with these three conditions, where we identify X_0 with G_0.

1. For all even j strictly less than i, $\Theta_{1,j}(F_2 \oplus X_j) = \Theta_{2,j}(E_2)$ and X_{j+1} is their common value.
2. For all odd j strictly less than i, $\Theta_{1,j}(F_1 \oplus X_j) = \Theta_{2,j}(E_1)$ and X_{j+1} is their common value.
3. For all j strictly less than i, $\Phi_{j+1}(D_1 \oplus X_{j+1}) = D_2$.

We can think of a verified sequence as just a finite sequence of numbers, the indices of the functionals. If $\langle (\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i \rangle$ is a verified sequence, then each X_{j+1} named above is a nontrivial element of (G_{j+1}). In the other direction, for every i, G_i is the last element of some verified sequence.

Theorem 2.5 *Suppose that* $\boldsymbol{p}$ *specifies the sequence* $\langle g_i : i \in \omega \rangle$ *in* $\mathfrak{D}$. *Let* P *be a representative of the join of representatives of* $\boldsymbol{p}$. *For* $S \subseteq \omega$, *the following conditions are equivalent.*

1. S *is* $\Sigma^0_2(P)$.
2. *There is a* Q *of degree* q *such that* $P \geq_{\mathfrak{D}} Q$ *and* $\boldsymbol{p}^\frown\langle q \rangle$ *specifies the subsequence* $\langle g_i : i \in S \rangle$.

Proof We begin with (1). Let us suppose that there is a Q of degree q such that $P \geq_{\mathfrak{D}} Q$ and $\boldsymbol{p}^\frown\langle q \rangle$ specifies the subsequence $\langle g_i : i \in S \rangle$. Then

$$i \in S \iff (\exists X)[G_i \geq_{\mathfrak{D}} X \text{ and } X \oplus D_1 \geq_{\mathfrak{D}} D_2 \text{ and } Q \geq_{\mathfrak{D}} X].$$

First, we can expand the right-hand side of the equivalence so that

$$i \in S \iff \text{There are } \Psi_1,\ \Psi_2, \text{ and } \Phi \text{ such that } \Psi_2(\Psi_1(G_i) \oplus D_1) = D_2 \text{ and } \Phi(Q) = \Psi_1(G_i),$$

where Ψ_1, Ψ_2, and Φ are $\mathscr{D}$-functionals. Second, the ideal below G_i is characterized by the recursion relations in Definition 2.2. So, saying that there is a Ψ_1 such that $\Psi_1(G_i)$ has a certain property can also be expressed as follows. We can say that there is a verified sequence ending with X_i such that X_i has the property in question. Equality between $\mathscr{D}$-Turing reductions is Π^0_1 relative to P; the existence of the X_js can be asserted by the existence of a vector of indices for $\mathscr{D}$-Turing reductions; and so, the reformulation of the right-hand side is a $\Sigma^0_2(P)$ property.

Part (2) of the proof of Theorem 2.5 is the construction of an appropriate Q. We will describe the strategies involved and then discuss a construction combining them.

Suppose that S is $\Sigma^0_2(P)$, and let R be a bounded formula such that for all i, $i \in S$ if and only if $\exists n \forall m R(i, n, m, P)$. For the moment, let us focus on satisfying the statement 'If $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is a verified sequence, then $Q \geq_{\mathscr{D}} X_i$ if and only if $i \in S$'.

Our strategy to satisfy this statement will have two types of substrategies which we will describe in isolation. Before we describe these, we discuss some mechanical preliminaries.

In the first type of substrategy, we will work with a number n and approximate whether $\forall m R(i, n, m, P)$. For this, we will do the first s-many computational steps of the process to check whether $R(i, n, 0, P)$, $R(i, n, 1, P)$, $R(i, n, 2, P)$, and so on. We say that $\forall m R(i, n, m, P)$ is verified up to stage s if and only if this s-step process does not reveal an m such that $R(i, n, m, P)$ is not true.

In both types of substrategy, we will approximate verified sequences. We let X_0 denote G_0. We say that a sequence $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is verified up to s provided that all of the equalities in the list

1. For all even j strictly less than i, $\Theta_{1,j}(F_2 \oplus X_j) = \Theta_{2,j}(E_2)$ and we let X_{j+1} denote common value.
2. For all odd j strictly less than i, $\Theta_{1,j}(F_1 \oplus X_j) = \Theta_{2,j}(E_1)$ and we let X_{j+1} denote their common value.
3. For all j strictly less than i, $\Phi_{j+1}(D_1 \oplus X_{j+1}) = D_2$.

hold on the set of computations that can be verified in s many steps. Clearly, $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ defines a verified sequence if and only if for every s it is verified up to s.

We are not being specific about what we mean by the set of computations that can be verified in s many steps. Our construction is not sensitive on this point. Fix any recursive method to eventually check through all of the above identities at all possible arguments, no matter how inefficient. Then take "the first s many steps" to mean the first s many steps in this recursive process.

We will describe our construction as occurring in stages. Stage s will consist of calculating for all σ of length s, whether σ is an element of Q. For each such σ, this calculation will have length a constant multiple of s^2. We note that membership in Q has been decided for strings of length less than s and that we can use information about Q on short strings provided that we can compute that information within our s^2 time bound.

Coding Fix the sequence $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ and a number n. The coding substrategy acts as follows to ensure that if $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is a verified sequence and $(\forall m)R(i, n, m, P)$ then $Q \geq_{\mathcal{D}} X_i$.

It chooses a finite binary string σ, unused by any other substrategy. Looking across all stages, its action breaks into two states.

State 1 If $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is verified up to s and no counterexample (m) to $(\forall m)R(i, n, m, P)$ is discovered within a search of s steps, then the coding substrategy acts as follows. Given a string $\sigma^\frown\tau^*$ of length s, it checks whether τ^* is the concatenation of a sequence of 0s of length the run time of the computation relative to the appropriate (depending on the parity of i) E_k used to determine whether $\tau \in X_i$, 1, and then τ. If this is the case, then it sets

$$\sigma^\frown\tau^* \in Q \iff \tau \in X_i.$$

Otherwise, it sets $\sigma^\frown\tau^* \notin Q$. Note that τ and $X_i(\tau)$ can be computed a constant multiple of s many steps from τ^* and the appropriate E_k.

State 2 If during stage s $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is not verified up to s or if in less than n-steps we discover an m such that $R(i, n, m, P)$, then the coding substrategy imposes the constraint that for all $\sigma^\frown\tau^*$ of length s, $\sigma^\frown\tau^* \notin Q$.

Effect If $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is a verified sequence, then the coding substrategy ensures that X_i is $\mathcal{D}$-computable from Q. Otherwise, its effect on the construction of Q is to ensure that there are only finitely many extensions of σ that belong to Q.

Diagonalization Our substrategy here is analogous to similar strategies found in Ladner [3]. Suppose that we are given a sequence $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$. The diagonalization substrategy acts to ensure that if the sequence is a verified sequence then $\Theta(Q) \neq X_i$. It affects the construction as follows.

State 1 First, it checks whether $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is verified up to s. If so then it runs the first s many steps to check whether there is a counterexample to $\Theta(Q) = X_i$. (Note that we restrict P's simulating queries to Q so that those queries are on arguments of length less than s.) If no counterexample is found, then the diagonalization substrategy requires that for all σ of length s, $\sigma \in Q$ if and only if σ is required to be in Q by virtue of a coding substrategy which has higher priority than it does. (We will organize our construction so that there are only finitely many strategies of higher priority than this one. We will also ensure that none of the strategies of higher priority code nontrivial sets $\mathcal{D}$-below G_i.)

State 2 If either $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is not verified up to s or the diagonalization discovers a counterexample to $\Theta(Q) = X_i$, then it imposes no constraint on the construction during stage s.

Effect The diagonalization substrategy starts by imposing a constraint that, if permanent, would ensure that Q is in the ideal generated from the sets coded by the higher priority coding substrategies. Further, if this constraint were permanent, then the verified sequence for X_i would ensure that X_i is a nontrivial element of (G_i) and the substrategy's never finding a counterexample to $\Theta(Q) = X_i$ would ensure that X_i is in the ideal generated by these coded sets. Consequently, if X_i is not below the join of the sets coded into Q by the action of higher priority coding substrategies, then the diagonalization strategy cannot stay in State 1 indefinitely.

The global strategy for $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ Now we discuss the global strategy $\mathscr{G}$ to ensure that if $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is a verified sequence, then $Q \geq_{\mathscr{D}} X_i$ if and only if $i \in S$. (Recall that R is a bounded formula such that for all i, $i \in S$ if and only if $\exists n \forall m R(i, n, m, P)$.)

The first action of $\mathscr{G}$ is to compute what we will call its *state* as follows. $\mathscr{G}$ first computes whether $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$ is verified up to s. If the sequence is not verified up to s, then we say that $\mathscr{G}$ is canceled during stage s and let *canceled* be its state during stage s. If it is not canceled, then beginning with n equal to 0, $\mathscr{G}$ performs the first s computational steps in the following recursion. (Below, we refer to a standard recursive enumeration of the $\mathscr{D}$-functionals $\langle\Theta_n : n \in \omega\rangle$.)

1. If there is an m less than s such that $\neg R(i, n, m, P)$ then $(n, 1)$ is not the state of $\mathscr{G}$.
2. If there is an x less than s such that $\Theta_n(x, Q) \neq X_i(x)$ then the state of $\mathscr{G}$ is not $(n, 2)$.

If $\mathscr{G}$ rules out both states $(n, 1)$ and $(n, 2)$ then it increases the value of n by one and repeats the process.

If $\mathscr{G}$ is not canceled, the state of $\mathscr{G}$ during stage s is the first pair (n, j) not ruled out above. It points to the substrategy that we should use for the sake of $\mathscr{G}$ during stage s.

If $\mathscr{G}$ is canceled, then it does not impose any constraints on the construction. Otherwise let (n, j) denote the state of $\mathscr{G}$ during stage s. If j is equal to 1, then $\mathscr{G}$ acts to enforce the state-1 constraints of the coding strategy on all strings τ of length s. If j is equal to 2, then $\mathscr{G}$ acts to enforce the state-1 constraints of the diagonalization strategy on all strings τ of length s.

Now we describe our full construction of Q.

Assigning priority Fix a recursive enumeration of all sequences of indices for possible verified sequences $\langle(\Theta_{1,j}, \Theta_{2,j}, \Phi_{j+1}) : j < i\rangle$. Of course, some of these may not actually denote verified sequences as they may fail to satisfy one of the appropriate equalities between terms. We let $\mathscr{G}_e$ denote the strategy associated with the eth such sequence (denoted $\langle(\Theta_{e,1,j}, \Theta_{e,2,j}, \Phi_{e,j+1}, X_{e,j+1}) : j < i_e\rangle$).

For distinct strategies $\mathscr{G}_{e_1}$ and $\mathscr{G}_{e_2}$, we say that $\mathscr{G}_{e_1}$ in state (n_1, k_1) has *higher priority* than $\mathscr{G}_{e_2}$ in state (n_2, k_2) if and only if either the maximum of $\{e_1, n_1\}$ is less than the maximum of $\{e_2, n_2\}$ or their maxima are equal and e_1 is less than e_2. Clearly, for each $\mathscr{G}_{e_1}$ and state (n_1, k_1), there are only finitely many $\mathscr{G}_{e_2}$s and states (n_2, k_2) which have higher priority.

Defining Q on sequences of length s During each stage s of our construction, we work through the following steps in order.

1. For each i less than s, we compute the state of $\mathscr{G}_i$ during stage s.
2. We order the strategies in their stage-s states according to the priority given above. Let $\mathscr{G}_e$ in stage-s state $(n_e, 2)$ have the highest priority among all of these whose states have the form $(n_j, 2)$ and for which there is no higher priority $\mathscr{G}_{e_1}$ in stage-s state $(n_{e_1}, 1)$ for which $i_e = i_{e_1}$. In other words, $\mathscr{G}_e$ in stage-s state $(n_e, 2)$ has the highest priority among those strategies/states $\mathscr{G}_{e_1}/(n_{e_1}, 2)$ for which there is no higher priority strategy/state which would code a nontrivial element of $(\mathscr{G}_{i_{e_1}})$ into Q.

3. Finally, if σ has length s, then σ is an element of Q if and only if some coding strategy $\mathcal{G}_{e_1}$ in a stage-s state of higher than or equal priority than that of $\mathcal{G}_e$ requires that σ belong to Q.

To summarize, we find the highest priority strategy/diagonalization-state pair $\mathcal{G}_e/(n_e, 2)$ which is working on a value of i for which there is no higher priority active coding strategy, and we restrict ourselves to using only those strategy/coding-state pairs of higher priority than $\mathcal{G}_e/(n_e, 2)$. We say that $\mathcal{G}_e/(n_e, 2)$ and these strategies in their coding states are active during stage s.

Verifying that Q has the requisite properties First, we observe that $P \geq_{\mathcal{D}} Q$. Suppose that σ is a string of length s. For each i less than s, we calculate the stage-s state of $\mathcal{G}_i$ by simulating two s-step computations, one to test whether to cancel $\mathcal{G}_i$ and one to determine its (n, j) state. Thus the calculation of the strategy/state pairs which are active during stage s is done in a constant multiple of s^2 many steps. We then determine whether σ is an element of Q by checking whether it is put into Q for a strategy which is active during stage s. As we indicated earlier, whether σ is to be put into Q by an active coding strategy can be determined in linear time relative to E_1 and E_2. Consequently, whether σ belongs to Q is computable from P in a constant multiple of s^2 many steps, and so $P \geq_{\mathcal{D}} Q$.

We now have a finite injury argument to show that Q satisfies a sufficient set of requirements. To begin, for each strategy and each state which that strategy can achieve, there are only finitely many strategy/state pairs of higher priority. Further, for all strategies $\mathcal{G}_e$ and states (n, j), if (n, j) is ruled out for $\mathcal{G}_e$ during stage s, then it is ruled out during every subsequent stage (for the same reason that it was ruled out during stage s).

We claim that no strategy $\mathcal{G}_e$ can reach a state $(n_e, 2)$ and remain actively in that state during all subsequent stages. For the sake of a contradiction, suppose that the claim is false. Let $\mathcal{G}_e$ and state $(n_e, 2)$ be the highest priority counterexample. Since $\mathcal{G}_e$ remains in state $(n_e, 2)$, its sequence $\langle(\Theta_{e,1,j}, \Theta_{e,2,j}, \Phi_{e,j+1}, X_{e,j+1}) : j < i_e\rangle)$ must be verified up to s during every stage s, and so be a verified sequence. In particular, X_{e,i_e} is a nontrivial member of (G_{i_e}). Choose a stage s so that $\mathcal{G}_e$ is in state $(n_e, 2)$ during stage s and so that every strategy in a stage-s state $(n, 2)$ with no active, higher priority, coding strategy has lower priority than $\mathcal{G}_e/(n_e, 2)$ does. By the choice of $\mathcal{G}_e$, $(n_e, 2)$, and s, if t is greater than or equal to s and if $\mathcal{G}_{e_1}$ is in a coding state of higher priority than $\mathcal{G}_e$ in state $(n_e, 2)$, then i_{e_1} is not equal to i_e. Consequently, the join of the sets being coded into Q by strategies of higher priority is below a finite join of G_is such that i is not equal to i_e. Since the degrees of the sets G_i are part of the sequence specified by $\boldsymbol{p}$ (see Definition 2.2), any set below the finite join above G_{i_e} has trivial $\mathcal{D}$-degree. Since X_{e,i_e} is the last element of a verified sequence, X_{e,i_e} is not trivial and $X_{e,i_e} \in (G_{i_e})$. Thus, X_{e,i_e} is not below the join of the coded sets. Once the construction computes the witness to this effect and rules out the state $(n_e, 2)$ for $\mathcal{G}_e$, as claimed.

By the previous paragraph, no strategy can remain in an active diagonalization state indefinitely. Suppose that $\langle(\Theta_{e,1,j}, \Theta_{e,2,j}, \Phi_{e,j+1}) : j < i_e\rangle$ is a verified sequence such that $i_e = i$ and that $(\forall n)(\exists m)\neg R(i, n, m)$. By the first assumption, $\mathcal{G}_e$ will have a state of the form (n, j) during every stage of the construction greater than or equal to e. We let n^* be fixed for the moment and we show that $\Theta_{n^*}(Q) \neq X_i$. Since the $(n, 1)$ states of any $\mathcal{G}_{e_1}$ with $i_{e_1} = i$ are discarded when the construction finds that $(\exists m)\neg R(i_e, n, m)$, no such $\mathcal{G}_{e_1}$ can be in one of these states indefinitely: either the construction discovers that $\langle(\Theta_{e_1,1,j}, \Theta_{e_1,2,j}, \Phi_{e_1,j+1}) : j < i_{e_1}\rangle$ fails to be verified up to the current stage or it discovers that $(\exists m)\neg R(i_e, n, m)$. So there is a stage t after which every $\mathcal{G}_{e_1}$ with $i_{e_1} = i_e$ rules out all of the states $(n_1, 1)$ of higher priority than that of $\mathcal{G}_e$ in state $(n^*, 2)$. Since no strategy can remain in an active diagonalization state indefinitely and no state can be repeated once it is ruled out, there is an even larger stage such that for all later stages, if $\mathcal{G}_e$ is in state $(n^*, 2)$ then it will be active. Since it cannot be active indefinitely, there must be a stage during which we rule out the state $(n^*, 2)$ for $\mathcal{G}_e$ and this can only happen by finding a string σ and a computation showing that $\Theta_{n^*}(\sigma, Q)$ is not equal to $X_{e,i}(\sigma)$. Consequently, if $(\forall n)(\exists m)\neg R(i, n, m)$ and $\langle(\Theta_{e,1,j}, \Theta_{e,2,j}, \Phi_{e,j+1}) : j < i_e\rangle$ is a verified sequence such that $i_e = i$, then $\Theta_{n^*}(Q) \neq X_i$. Since n^* was arbitrary, if $(\forall n)(\exists m)\neg R(i, n, m)$ and $\langle(\Theta_{e,1,j}, \Theta_{e,2,j}, \Phi_{e,j+1}) : j < i_e\rangle$ is a verified sequence such that $i_e = i$ then $Q \not\geq_{\mathcal{D}} X_i$, as required.

Dually, suppose that $\langle(\Theta_{e,1,j}, \Theta_{e,2,j}, \Phi_{e,j+1}) : j < i_e\rangle$ is a verified sequence such that $i_e = i$ and that $(\exists n)(\forall m)\neg R(i, n, m)$. Let n_i be the smallest number n such that $(\forall m)\neg R(i, n, m)$. Arguing as above, either there is a stage s such that all of the higher priority states of $\mathcal{G}_e$ are ruled out during every stage after s or there is a higher priority strategy/coding-state $\mathcal{G}_{e_1}$ which is active indefinitely and for which $i_{e_1} = i$. All of the strategy/diagonalization-states of higher priority than $\mathcal{G}_e$ in state $(n_i, 1)$ which are ever active are eventually ruled out. If $\mathcal{G}_e$ is ever made active in state $(n_i, 1)$ it will be active during every subsequent stage. Consequently, either $\mathcal{G}_e$ in state $(n_i, 1)$ is active indefinitely or there is a higher priority strategy/coding-state $\mathcal{G}_{e_1}$ which is active indefinitely and for which $i_{e_1} = i$. (Note that the coding of X_{e_1} into Q could keep $\mathcal{G}_e$ in an earlier diagonalization state. However, in this case, we need not argue that $\mathcal{G}_e$ codes $X_{e,i}$ into Q.)

Consequently, there is an e^* and a verified sequence ending with $X_{e^*,i}$ such that $Q \geq_{\mathcal{D}} X_{i_{e^*}}$, as required in this case. Thus Q satisfies the requirements necessary to verify part (2) of Theorem 2.5. □

Next we show that it is possible for parameters to specify sequences. The following theorem is not the best possible, in fact stronger results appear in [7], but it is sufficient for our application. We include a direct proof of Theorem 2.6 here, since it is relatively short and avoids the complexities of [7] that are not relevant here.

Theorem 2.6 *There are sets E_1, F_1, E_2, F_2, D_1, and D_2 such that the following conditions hold.*

1. *The degrees of E_1, F_1, E_2, F_2, D_1, and D_2 specify a sequence.*
2. *The Turing jump of the join of all of these sets is recursive in the Turing jump of the least element of $\mathcal{D}$.*

Proof Recall our notation, O is a representative of the least element of $\mathcal{D}$. We build E_1, F_1, E_2, F_2, D_1, and D_2 by an effective forcing construction so that the Turing jump of their join is recursive in O'.

We partition the set of finite binary strings with at least one nonzero value into an infinite set of isomorphic copies of the set of all binary strings. Let $\langle 0^i 1\rangle$ be the binary sequence with i-many 0s followed by a 1. For a set X, we let X^i denote the set of strings σ such that σ is in X and σ is an extension of $\langle 0^i 1\rangle$. We will let $G_{2j} = E_1^{2j}$ and $G_{2j+1} = E_2^{2j+1}$, and so by specifying E_1 and E_2 we will implicitly specify all of the elements of $\{G_i : i \in \omega\}$ as well. Note that we have ensured that distinct G_is have empty intersection and that the set theoretic union of any set of G_is is a *PTIME*-upper bound for the ideal which they generate.

The forcing partial order A condition p in P specifies finitely much about the sets E_1, F_1, E_2, F_2, D_1, and D_2. The information specified must satisfy the following conditions.

1. (a) If p specifies $D_2(\sigma)$, $G_i(\langle 0^i 1\rangle^\frown\sigma)$, and $D_1(\langle 0^i 1\rangle^\frown\sigma)$, then
$$D_2(\sigma) = \begin{cases} 0, & \text{if } D_1(\langle 0^i 1\rangle^\frown\sigma) = G_i(\langle 0^i 1\rangle^\frown\sigma); \\ 1, & \text{otherwise.} \end{cases}$$
   (b) Further, if p specifies two of the above three values, then it specifies the remaining one.

2. (a) If $i > 0$ is odd and p specifies $G_i(\langle 0^i 1\rangle^\frown\sigma)$, $F_1(\langle 0^i 1\rangle^\frown\sigma)$, and $G_{i+1}(\langle 0^{i+1} 1\rangle^\frown\sigma)$, then
$$G_{i+1}(\langle 0^i 1\rangle^\frown\sigma) = \begin{cases} 0, & \text{if } F_1(\langle 0^i 1\rangle^\frown\sigma) = G_i(\langle 0^i 1\rangle^\frown\sigma); \\ 1, & \text{otherwise.} \end{cases}$$
   (b) If $i > 0$ is even and p specifies $G_i(\langle 0^i 1\rangle^\frown\sigma)$, $F_2(\langle 0^i 1\rangle^\frown\sigma)$, and $G_{i+1}(\langle 0^{i+1} 1\rangle^\frown\sigma)$, then
$$G_{i+1}(\langle 0^{i+1} 1\rangle^\frown\sigma) = \begin{cases} 0, & \text{if } F_2(\langle 0^i 1\rangle^\frown\sigma) = G_i(\langle 0^i 1\rangle^\frown\sigma); \\ 1, & \text{otherwise.} \end{cases}$$
   (c) In each of the above cases, if p specifies two of the above three values, then it specifies the remaining one.

Conditions are ordered by inclusion.

Properties of a generic set The instances of comparability required to specify a sequence are built into the partial order. Additionally, there is enough flexibility in the partial order so that the other properties required by Definition 2.2 can be ensured by deciding $\Sigma^0_1(O)$ sentences about the sets constructed. Specifically, we must satisfy the following requirements.

***D*-requirements** For each $\mathcal{D}$-functional Θ, $\Theta(D_1) \neq D_2$.
***FE*-requirements** For each pair of $\mathcal{D}$-functionals Φ_1 and Φ_2 and each i,
(a) if i is odd and $\Phi_1(F_1G_i) = \Phi_2(E_1)$, then there is a $\mathcal{D}$-functional Δ such that $\Delta(G_{i+1}) = \Phi_2(E_1)$, and
(b) if i is even and $\Phi_1(F_2G_i) = \Phi_2(E_2)$, then there is a $\mathcal{D}$-functional Δ such that $\Delta(G_{i+1}) = \Phi_2(E_1)$.

***D*-requirements** Consider the first of these requirements. Suppose that p is a condition. Choose σ so that p does not specify $D_2(\sigma)$, and for all i, p does not specify $G_i(\langle 0^i 1\rangle^\frown\sigma)$ or $D_1(\langle 0^i 1\rangle^\frown\sigma)$. We first extend p's specification of D_1 so as to determine the value of $\Theta(\sigma, D_1)$. Then we extend p's specification of D_2 so that $D_2(\sigma)$ is different from the value of $\Theta(\sigma, D_1)$. We extend the specification of the G_is to ensure that $D_1 \oplus G_i$ codes D_2 in the manner prescribed in (1a). Finally we extend F_1 and F_2 so as to respect (2a) and (2b). Given p, we can find the desired extension recursively in O.

***FE*-requirements** Now consider the first instance of the second requirement, when i is odd. Suppose that p is a condition. By making a finite extension of p, we may assume that for all σ and all i, if either of $F_1(\langle 0^i 1\rangle^\frown\sigma)$ or $F_2(\langle 0^i 1\rangle^\frown\sigma)$ is specified by p, then so are $G_i(\sigma)$, and $G_{i+1}(\sigma)$. We consider two cases.

Case 1 For any string σ and any two conditions q_1 and q_2 which extend p and agree on the values specified for G_{i+1}, the values of $\Phi_2(\sigma, E_1)$ determined by the conditions are equal.

But then, for any way to extend the values of p onto the G_js, there is an extension of the values of F_1, F_2, D_1, and D_2 which produces a condition. Consequently p forces that G_{i+1} can compute the value of $\Phi_2(\sigma, E_1)$: the value of $\Phi_2(\sigma, E_1)$ is equal to that of $\Phi_2(\sigma, E_p)$, where E_p is the set whose only elements are the union of G_{i+1} with the set of elements specified to belong to E_1 by p.

Case 2 There is a string σ and two conditions q_1 and q_2 which extend p and agree on the values specified for G_{i+1} such that the values of $\Phi_2(\sigma, E_1)$ determined by the conditions are not equal, and we fix such.

By making a finite extension of q_1, we may assume that q_1 specifies enough of F_1 and G_i to determine the value of $\Phi_1(\sigma, F_1 \oplus G_i)$. If this value is different from $\Phi_2(E_1)$, then our requirement is satisfied. Otherwise, we proceed as follows to construct a condition r such that r specifies the same values for F_1 and G_i as q_1 does, and r specifies the values for E_1 that q_2 does. We start with q_1. We change the values specified for E_1 so as to agree with those specified by q_2; since q_1 and q_2 specify the same values for G_{i+1}, this does not change the specification of G_{i+1}. Thus we have changed the specification of some of the G_js with j even. We change the values of the G_js for j odd in order to make $F_1 \oplus G_j$ correctly code the new values of G_{j+1}. Note that though we may change some of the G_js for j even, we do not change G_i, since $F_1 \oplus G_i$ already codes G_{i+1}. We change the values of F_2 in order to make $F_2 \oplus G_j$ correctly code the new values of the G_{j+1}s for j even. Finally, we change D_1 so that for all j, $D_1 \oplus G_j$ correctly codes D_2. In short, we can change all of the even G_js with $j \neq i+1$ and shunt the feedback away for $F_1 \oplus G_i$. The condition r ensures that $\Phi_1(\sigma, F_1 \oplus G_i) \neq \Phi_2(\sigma, E_1)$.

In either case, the requirement is satisfied. The split into cases is $\Sigma^0_1(O)$; and in the second case, we can find the condition r uniformly recursively in O'.

Turing jump requirements Controlling the jump is a standard feature of constructions of this sort; see Jockusch [2]. We can ensure that the join of the sets that we produce has Turing jump recursive in O' by ensuring that every Σ^0_1 sentence about these sets is decided by an element of our partial order.

The construction For each of our requirements, we can go from a condition p to a condition q such that the requirement is satisfied for any sets extending q. Further, we can find q from p and the requirement recursively relative to O'. Consequently, we can use recursion to construct sets of the desired sort, satisfying the requirements one after the other. $\square$

2.2 Comparing $\mathcal{D}_{tt}(\geq_{tt}\mathbf{0}')$ and $\mathcal{D}_{PTIME}$

Theorem 2.7 *$\mathcal{D}_{tt}(\geq_{tt}0')$ and $\mathcal{D}_{PTIME}$ are not isomorphic.*

Proof First consider $\mathcal{D}_{tt}(\geq_{tt}0')$. Every element of $\mathcal{D}_{tt}(\geq_{tt}0')$ can compute $0'$, and so $0'''$ is Σ^0_2 in every representative of an element of $\mathcal{D}_{tt}(\geq_{tt}0')$. By Theorem 2.5 , if

$\boldsymbol{p}$ specifies a sequence $\langle g_i : i \in \omega\rangle$ in $\mathcal{D}_{tt}(\geq_{tt} 0')$, then there is a q below the join of $\boldsymbol{p}$ such that $\boldsymbol{p}^\frown\langle q\rangle$ specifies the subsequence $\langle g_i : i \in 0'''\rangle$.

Now consider $\mathcal{D}_{PTIME}$. Let $\boldsymbol{p}$ be the sequence of parameters produced in Theorem 2.6. By clause (1) of Theorem 2.6, $\boldsymbol{p}$ specifies a sequence $\langle g_i : i \in \omega\rangle$ in $\mathcal{D}_{PTIME}$. By clause (2), the Turing jump of join of the representatives of $\boldsymbol{p}$ is recursive in $0'$. Consequently, if q is below the join of $\boldsymbol{p}$ in $\mathcal{D}_{PTIME}$ and Q is a representative of Q, then any set $\Sigma^0_2(Q)$ is Σ^0_2. But then, since $0'''$ is not Σ^0_2, Theorem 2.5 implies that the subsequence $\langle g_i : i \in 0'''\rangle$ is not represented by any q below the join of the elements of $\boldsymbol{p}$. □

3 Conclusion

We have shown that $\mathcal{D}_{tt}(\geq_{tt} 0')$ and $\mathcal{D}_{PTIME}$ are not isomorphic. We firmly believe that these structures have different first-order theories. We believe that one could find a difference between their theories by extending the apparatus of specifying sequences to an apparatus of specifying standard models of arithmetic. The structural difference between the two structures would then be expressed in the first-order language of these structures. $\mathcal{D}_{PTIME}$ would have a sequence of parameters specifying a standard model of arithmetic such that no q below the parameters specifies the complete Σ^0_3 predicate on that model. In $\mathcal{D}_{tt}(\geq_{tt} 0')$, the opposite would be true. One could attempt to apply the techniques in [5] in order to carry out this proposal.

The difference found between $\mathcal{D}_{tt}(\geq_{tt} 0')$ and $\mathcal{D}_{PTIME}$ comes from the large difference in the Turing degrees of their least elements. Our methods do not answer the following question.

Question 3.1 *Let $\mathcal{D}_{ELEM}$ be the elementary-time Turing degrees. Is $\mathcal{D}_{PTIME}$ isomorphic to $\mathcal{D}_{ELEM}$?*

References

[1] Ambos-Spies, K., “Minimal pairs for polynomial time reducibilities,” pp. 1–13 in *Computation Theory and Logic*, vol. 270 of *Lecture Notes in Computer Science*, Springer, Berlin, 1987. Zbl 0637.03038. MR 88j:03025. 3

[2] Jockusch, C. G., Jr., “Degrees of generic sets,” pp. 110–39 in *Recursion Theory: Its Generalisation and Applications (Proceedings of the Logic Colloquium, University of Leeds, Leeds, 1979)*, edited by F. R. Drake and S. S. Wainer, vol. 45 of *London Mathematical Society Lecture Note Series*, Cambridge University Press, Cambridge, 1980. Zbl 0457.03042. MR 83i:03070. 10

[3] Ladner, R., “On the structure of polynomial time reducibility,” *Journal of the Association for Computing Machinery*, vol. 22 (1975), pp. 155–71. Zbl 0322.68028. MR 57:4623. 5

[4] Nies, A., R. A. Shore, and T. A. Slaman, “Interpretability and definability in the recursively enumerable degrees,” *Proceedings of the London Mathematical Society. Third Series*, vol. 77 (1998), pp. 241–91. Zbl 0904.03028. MR 99m:03083. 2

[5] Shinoda, J., and T. A. Slaman, “On the theory of the PTIME degrees of the recursive sets,” *Journal of Computer and System Sciences*, vol. 41 (1990), pp. 321–66. Zbl 0715.68040. MR 92b:03049. 3, 11

[6] Shore, R. A., "The theory of the degrees below $0'$," *The Journal of the London Mathematical Society. Second Series*, vol. 24 (1981), pp. 1–14. Zbl 0469.03027. MR 83m:03051. 2

[7] Shore, R. A., and T. A. Slaman, "The p-T-degrees of the recursive sets: Lattice embeddings, extensions of embeddings and the two-quantifier theory," *Theoretical Computer Science*, vol. 97 (1992), pp. 263–84. Zbl 0774.03028. MR 93e:03061. 3, 8

Acknowledgments

During the preparation of this paper, Slaman was partially supported by the Alexander von Humboldt Foundation and by National Science Foundation Grant DMS-9988644.

Department of Informatics
Athens University of Economics and Business
Patission 76
10434 Athens
GREECE
xar@aueb.gr

Department of Mathematics
University of California at Berkeley
719 Evans Hall #3840
Berkeley CA 94720-3840
slaman@math.berkeley.edu