ELSEVIER

Annals of Pure and Applied Logic 84 (1997) 139–152

ANNALS OF PURE AND APPLIED LOGIC

Automorphisms in the *PTIME*-Turing degrees of recursive sets [1]

Christine Ann Haught [a,b,*], Theodore A. Slaman [a]

[a] *Department of Mathematics, The University of Chicago, Chicago, IL 60637, USA*

[b] *Department of Mathematical Sciences, Loyola University of Chicago, Chicago, IL 60626, USA*

Received 22 November 1993

Communicated by A. Nerode

Abstract

We consider questions related to the rigidity of the structure $\mathscr{R}$, the *PTIME*-Turing degrees of recursive sets of strings together with *PTIME*-Turing reducibility, $\leqslant_{pT}$, and related structures; do these structures have nontrivial automorphisms? We prove that there is a nontrivial automorphism of an ideal of $\mathscr{R}$. This can be rephrased in terms of partial relativizations. We consider the sets which are *PTIME*-Turing computable from a set A, and call this class $PTIME^A$. Our result can be stated as follows: There is an oracle, A, relative to which the *PTIME*-Turing degrees are not rigid (i.e. there is a nontrivial automorphism of the structure $\langle PTIME^A, \leqslant_{pT} \rangle$). Furthermore, the automorphism can be made to preserve the complexity classes $DTIME^A(n^k)$ (the collection of sets computable from A by a deterministic computation with time bound of order n^k) for all $k \geqslant 1$, or to move any $DTIME^A(n^k)$ for $n \geqslant 2$. From the existence of such an automorphism we conclude as a corollary that there is an oracle A relative to which the classes $DTIME(n^k)$ are not definable over $\mathscr{R}$. We carry out the corresponding partial relativization for the many-one degrees to construct an oracle, A, relative to which the *PTIME*-many-one degrees relative to A have a nontrivial automorphism, and one relative to which the lattice of sets in $PTIME^A$ under inclusion have a nontrivial automorphism. The proof is phrased as a forcing argument; we construct the set A to meet a particular collection of dense sets in our notion of forcing. Roughly, the dense sets will guarantee that if A meets these sets and we split A into two pieces, A_0 and A_1, in a simple way, and then switching the roles of A_0 and A_1 in all computations from A will produce an automorphism of the ideal of *PTIME*-degrees below A. We force A_0 and A_1 to have different *PTIME*-Turing degree; this will then make the automorphism nontrivial. An appropriately generic set A is constructed using a priority argument.

Keywords: Polynomial-time degrees; Automorphisms

AMS subject classifications: 03D15; 03D30; 03D80; 68Q15

* Corresponding address: Department of Mathematics, The University of Chicago, Chicago, IL 60637, USA

[1] The authors' research was supported by NSF grants DMS-8705818 and DMS-8601856, respectively. In addition, Slaman was supported by Presidential Young Investigator Award DMS-8451748.

SSDI 0168-0072(95)00065-8

1. Introduction

A recursive set, R, is one for which there is an effective method for determining membership in R. A set X is recursive in, or Turing computable from, Y if there is an effective method which, when given complete information about Y, determines all membership information about X. If X and Y are each recursive in the other, then X and Y have the same Turing degree. The recursive sets are those which are, in theory, computable. Turing computability is one notion of relative complexity among sets. If we wish to study relative complexity among computable sets then a finer notion is needed since all recursive sets are of the same Turing degree. We consider recursive sets of strings. In general, a string is a finite sequence of symbols from a finite alphabet. In this paper, all strings are binary, i.e. strings over the alphabet $\{0,1\}$. Usually, strings are denoted by lower case greek letters near the end of the alphabet, σ, τ, μ. We use $|\sigma|$ to denote the length of σ. Sets of strings are denoted by upper-case Roman letters. The finer notion of relative complexity we use is Turing computability with polynomial-time bounds, *PTIME*–computability, as defined in [2]. A *PTIME* reduction Φ is a Turing reduction and a polynomial φ such that for any oracle X and any string σ, $\Phi(X)(\sigma)$ (the reduction procedure with oracle X evaluated at σ) can be evaluated in less than or equal to $\varphi(|\sigma|)$ steps. Notice that the time bound for the computation is the same regardless of the choice of oracle set. The particular model of computation used is not important; we only use the fact that the computation is deterministic and the time bounds are closed under composition. We use upper-case greek letters to represent *PTIME*-Turing reductions, and the corresponding lower-case greek letters to represent the corresponding polynomial-time bounds.

We say that the set A is *PTIME* computable from B ($A \leqslant_{pT} B$) if there is a *PTIME* reduction Φ such that for all strings σ, $A(\sigma) = \Phi(B)(\sigma)$. A and B are of the same *PTIME*-Turing degree if $A \leqslant_{pT} B$ and $B \leqslant_{pT} A$.

We are interested in the following structures:

$$
\begin{aligned}
\mathscr{R} &= \langle \textit{PTIME}\text{-Turing degrees of recursive sets}, \leqslant_{pT} \rangle \\
\mathscr{R}(\leqslant_{pT} \mathbf{a}) &= \langle \textit{PTIME}\text{-Turing degrees} \leqslant_{pT} \mathbf{a}, \leqslant_{pT} \rangle \\
\textit{PTIME}^A &= \textit{PTIME} \text{ relativized to } A \\
&= \{B : B \leqslant_{pT} A\} \\
\mathscr{P}^A &= \langle \textit{PTIME} \text{ degrees of } \textit{PTIME}^A, \leqslant_{pT} \rangle = \mathscr{R}(\leqslant_{pT} \mathbf{a}) \\
\textit{DTIME}(n^k) &= \{B: \text{there is a } \textit{PTIME} \text{ reduction } \Phi \text{ with time bound } cn^k, \\
&\qquad \text{where } c \text{ is a constant, such that } B = \Phi(\emptyset)\} \\
\textit{DTIME}^A(n^k) &= \textit{DTIME}(n^k) \text{ relativized to } A \\
&= \{B: \text{there is a } \textit{PTIME} \text{ reduction } \Phi \text{ with time bound } cn^k, \\
&\qquad \text{where } c \text{ is a constant, such that } B = \Phi(A)\} \\
\mathscr{L}_A &= \langle \textit{PTIME}^A, \subseteq \rangle \\
\mathscr{L}_A^* &= \langle \textit{PTIME}^A, \subseteq^* \rangle
\end{aligned}
$$

where $X \subseteq^* Y$ if for all but finitely many strings σ, $\sigma \in X$ implies $\sigma \in Y$.

We show that some of these structures are not rigid, i.e. that there are nontrivial automorphisms of the structures.

Theorem 1. *There is a recursive set A such that there is a nontrivial automorphism of $\mathscr{R}(\leqslant_{pT} \mathbf{a})$ (where $\mathbf{a}$ = PTIME Turing degree of A).*

This can be rephrased as a statement about the degrees in a relativized *PTIME* class.

Theorem 1′. *There is a recursive oracle A such that $\mathscr{P}^A$ is not rigid.*

This can be strengthened to show that there is a recursive oracle A, with a nontrivial automorphism which fixes all of the classes $DTIME^A(n^k)$ and that there is a recursive oracle A, with a nontrivial automorphism which moves an arbitrary class $DTIME^A(n^k)$. From this we conclude that $DTIME^A(n^k)$ is not definable over $\mathscr{P}^A$.

Theorem 2. (1) *There is a recursive oracle, A, and a nontrivial automorphism of $\mathscr{P}^A$, induced by a map f on $PTIME^A$ such that for all natural numbers $k \geqslant 1$, $f(DTIME^A(n^k)) = DTIME^A(n^k)$.*

(2) *For any natural numbers $k, j \geqslant 1$ there is a recursive oracle, A, and a nontrivial automorphism of $\mathscr{P}^A$, induced by a map f on $PTIME^A$ such that for some $B \in DTIME^A(n^k)$, $f(B) \in DTIME^A(n^j)$, and for all $l < k, B \notin DTIME^A(n^l)$, and for all $m < j, f(B) \notin DTIME^A(n^m)$.*

Corollary. *For each $k \geqslant 1$ there is a recursive oracle A such that the class $DTIME^A(n^k)$ is not definable over $\mathscr{P}^A$.*

The theorem can be further strengthened to show that there is a recursive oracle A relative to which there is a map on $PTIME^A$ which simultaneously induces a nontrivial automorphism of the degree structure $\mathscr{P}^A$ and of the lattice of sets *PTIME* in A.

Theorem 3. *There is a recursive oracle A, and a map $g : PTIME^A \to PTIME^A$ such that g is an automorphism of $\mathscr{L}_A$ and g induces an automorphism of $\mathscr{P}^A$.*

The proofs of Theorems 2 and 3 involve only slight modifications and observations about the proof of Theorem 1. Thus, we devote most of the paper to the proof of Theorem 1. Briefly, the proof of Theorem 1 involves the construction of a generic set A, which can be decomposed into two sets $A = A_0 \oplus A_1$. The automorphism is obtained by interchanging the roles of A_0 and A_1 in *PTIME* reductions from A. We can then see that the *PTIME* degrees below the degree of A_0 get mapped to the *PTIME* degrees below that of A_1. For Theorem 2, we analyze the decomposition of A into A_0 and A_1, or dually, the embedding of A_0 and A_1 into A. In the original proof the embedding can be calculated in linear-time, and this is enough to conclude that the classes $DTIME^A(n^k)$ are preserved. We notice that this embedding can also be modified so that A_0 is in,

say, $DTIME^A(n^k)$ and A_1 is in $DTIME^A(n^j)$. The automorphism then takes $deg(A_0)$ to $deg(A_1)$, and thus moves the class $DTIME^A(n^k)$.

A slight modification of the proof of Theorem 1 produces a map which is an automorphism of $\mathscr{L}_A^*$. Then, as in [5], we can show that the automorphism of $\mathscr{L}_A^*$ is in fact induced by a permutation of the set of all binary strings. This permutation simultaneously induces an automorphism of $\mathscr{L}_A$ and of $\mathscr{P}^A$.

In Section 2 we give an outline of the proof of Theorem 1 and discuss four key ideas in the proof. In Section 3 we give the construction of A. In Section 4 we define the map on $PTIME^A$ and verify that it induces an automorphism. In Section 5 we further discuss the proofs of Theorems 2 and 3.

2. Outline of the proof of Theorem 1

2.1. Bounded reducibility on recursive sets

We make two general observations about the kinds of objects we are studying. First, consider the bounded nature of *PTIME*-Turing computations. For any *PTIME* reduction Φ, any oracle set A, and any string σ, the computation $\Phi(A)(\sigma)$ can be evaluated in $\varphi(|\sigma|)$ steps. Thus, we may assume that all computations converge, since divergence can be treated as simply another value. In contrast with the Turing degrees in general, in the *PTIME*-Turing degrees "$\Phi(A) = B$" is a Π_1^0 statement about A and B.

Second, we are considering *PTIME* reductions applied to recursive sets. So, during a *PTIME* computation with recursive oracle A, we have the ability to simulate A, and to recognize that the simulation is correct. Suppose we are building a *PTIME* reduction Φ, and defining its value at $\Phi(A)(\sigma)$. We have the ability to run the algorithm for computing A (this exists because A is recursive) for polynomial in $|\sigma|$ many steps. Any answer given by this simulation must be correct information about A. As the length of σ increases we gain ability to simulate more and more of the oracle set. This ability to rely upon correct answers from the simulation of the oracle is in contrast with other kinds of approximations familiar from work in the Turing degrees, in particular recursive enumerations and Δ_2^0 approximations. In those approximations one must take into account the possibility that the approximation might change at a later stage. In our situation, we know that once a value appears it will never change, and that a value is guaranteed to appear eventually.

Fix an effective well-ordering, $\leqslant$, of $\{0,1\}^*$ which respects lengths; i.e. for $\sigma, \tau \in \{0,1\}^*$, if $|\sigma| < |\tau|$, then $\sigma < \tau$. A *condition* p is a function from $\{0,1\}^*$ to $\{0,1\}$ whose domain is finite and closed downward under this effective well-ordering of $\{0,1\}^*$. The *length* of a condition p, denoted by $|p|$, is the length of the longest string in the domain of p. For conditions p and q, we say that p *extends* q ($p \supseteq q$) if p contains more information than q does; i.e., for every string σ in the domain of q, σ is in the domain of p and $p(\sigma) = q(\sigma)$. We use conditions as approximations, or

initial segments of oracle sets. When A is a recursive set we use $p_{A,l}$ to denote the condition which represents the information about A which can be computed with l many steps. By this we mean that $A \upharpoonright \mathrm{domain}(p_{A,l})$ can be computed with l many steps. The number of steps it takes to compute A will be calculated using the rate at which A is built during our construction.

2.2. *Organization of the proof and definition of a generic set*

We will build a set S and a map Δ on *PTIME* reductions so that for each *PTIME* reduction Θ, $\Delta(\Theta)$ is a *PTIME* reduction. For notational convenience, we use Θ^* to denote $\Delta(\Theta)$. The map on *PTIME* reductions will induce a nontrivial automorphism of $\mathscr{P}^A$. The map is defined, roughly, by effectively splitting the oracle set A into two pieces A_0 and A_1, of different *PTIME* degree, and then switching the roles of A_0 and A_1 in computations from A. More specifically, we guarantee that Δ induces an automorphism of $\mathscr{P}^A$ by making the following three properties hold:

(1) $\Theta(A) \leqslant_{pT} \Psi(A)$ implies $\Theta^*(A) \leqslant_{pT} \Psi^*(A)$. (I.e. if there is a *PTIME* reduction Φ such that $\Phi(\Psi(A)) = \Theta(A)$, then there is a *PTIME* reduction $\widehat{\Phi}$ such that $\widehat{\Phi}(\Psi^*(A)) = \Theta^*(A)$.)

(2) $(\Theta^*)^*(A) = \Theta(A)$.

(3) For some *PTIME* reduction Θ, $\Theta(A) \not\equiv_{pT} \Theta^*(A)$.

To accomplish these goals we use a variation of the "looking back" techniques used by Ladner in his construction of a minimal pair of *PTIME* degrees [3]. In particular, to establish (1), we try whenever possible to ensure that

$$\Theta(A) \neq \Phi(\Psi(A))$$

by using a finite initial segment of A to force

$$\Theta(A)(\sigma) \neq \Phi(\Psi(A))(\sigma)$$

for some $\sigma \in 2^{<\omega}$. As in Ladner's construction, it is possible to construct a recursive A so that if it is impossible to force

$$\Theta(A)(\sigma) \neq \Phi(\Psi(A))(\sigma),$$

then by looking back, we can recover enough of the computations $\Theta(A)(\sigma)$ and $\Phi(\Psi(A))(\sigma)$ to be able to compute the value of $\Theta^*(A)(\sigma)$ from $\Psi^*(A)$ in a *PTIME*-Turing way.

In order to make the looking back strategy succeed, we need to spread out the information in the set A rather thinly, so that if the string σ is a potential element of A, then by virtue of its length alone, σ is large enough to recover the values of $A(\eta)$ for all potential elements, η of A of length less than the length of σ. Such "thinly distributed" sets have been used by Ladner [3] and Ambos-Spies [1], among others. We will call a set which possesses the version of thinness which is appropriate for our construction "ultra-sparse".

When we define ultra-sparseness formally we define it for A and take into account the coding of A_0 and A_1 into A. We fix the following coding of A_0 and A_1 into A:

$$A_0 = \{\sigma : \sigma \in A \text{ and } \sigma \text{ ends in } 0\},$$

$$A_1 = \{\sigma : \sigma \in A \text{ and } \sigma \text{ ends in } 1\}.$$

We now give the definition of ultra-sparseness appropriate for our construction. We use $\sigma \star \tau$ to denote the string which is the concatenation of the strings σ and τ.

Definition. A set of strings A is *ultra-sparse* if there is a linear-time computable set of strings, S, such that

(1) $\forall \tau \in \{0,1\}^*(A(\tau) = 1 \Rightarrow \exists \sigma \in S(\tau = \sigma \star 0 \text{ or } \tau = \sigma \star 1))$.

(2) $\forall n(S$ contains at most one string of length $n)$.

(3) If τ is the $(t+1)$st element of S (where the elements are ordered by length) and the tth element of S has length n, then in $\leqslant |\tau|$ many steps $A(\sigma)$ can be computed, for all σ such that $|\sigma| \leqslant n$.

Ambos-Spies [1] has used a condition similar to the above definition of ultra-sparse to show, among other things, that $\mathscr{R}$ is inhomogeneous.

As an illustration of the usefulness of this definition we prove the following proposition, which will be used later in the verification that our construction succeeds.

Proposition 1 (Ambos-Spies [1]). *If A is ultra-sparse and $B \leqslant_{pT} A$ then B can be computed from A by a computation which consults its oracle for at most two queries.*

Proof. Suppose that A is ultra-sparse via the linear-time computable set S. We use the following terminology: $\tau \in \{0,1\}^*$ is a *decision point* for A if and only if $\tau = \sigma \star i$, for some $\sigma \in S$ where $i = 0$ or 1.

Since $B \leqslant_{pT} A$, let Θ be a polynomial-time bounded Turing reduction such that $\Theta(A) = B$. We give a new computation of B from A based on Θ.

To compute $B(\eta)$: Let σ be the largest string in S such that $|\sigma| < \theta(|\eta|)$. Then $\sigma \star 0$ and $\sigma \star 1$ are the largest decision points in A relevant to the computation of $\Theta(A)(\sigma)(=B(\sigma))$. Simulate A for $|\sigma|$ many steps. This simulation determines $A(\tau)$ for all τ such that $|\tau| \leqslant |\sigma|$. Next simulate the computation of $\Theta(A)(\sigma)$, but when the computation attempts to consult the oracle with a query τ, and $|\tau| \leqslant |\sigma|$, then consult the simulation of A instead. If the computation queries the oracle at $\sigma \star 0$ or $\sigma \star 1$, then allow the oracle query to occur.

Thus, the only possible oracle queries in this computation are "$\sigma \star 0 \in A$?" or "$\sigma \star 1 \in A$?" Notice that the time bound for this new computation is a polynomial of degree no higher than the degree of θ. □

2.3. Notation and definition of the map Δ

For $l \in \omega$, the condition $p_{A,l}$ represents the information about A which can be determined in l many computation steps, subject to the provision that for all $\eta \in S$, and all $i = 0, 1$

$$p_{A,l}(\eta \star 1) \text{ is defined} \iff p_{A,l}(\eta \star (1-i)) \text{ is defined.}$$

For a condition q, a *PTIME*-Turing reduction Θ and $\sigma \in \{0,1\}^*$, if q is defined on all strings of length $\leqslant \theta(|\sigma|)$, then $\Theta(q)(\sigma)$ is the value computed by using the value of $q(\tau)$ as the answer to all oracle queries of the form "$\tau \in A$?" in the computation $\Theta(A)(\sigma)$.

We describe the computation of $\Theta^*(X)(\sigma)$, where X is an arbitrary oracle set and σ is a string.

To compute $\Theta^*(X)(\sigma)$, first compute $p_{A,|\sigma|}$. We simulate the computation done by $\Theta(A)(\sigma)$. When the computation for $\Theta(A)(\sigma)$ queries its oracle at a string τ for which $p_{A,|\sigma|}(\tau)$ is defined, then answer the query with the value $p_{A,|\sigma|}(\tau)$. For other queries, τ, write τ as $\tau = \mu \star i$, where $i = 0$ or 1, and replace the query "$\mu \star i \in A$?" by the query "$\mu \star (1-i) \in X$?"

Note that in this computation, the only oracle queries are ones of the form "$\mu \star i \in A$?" where $p_{A,|\sigma|}(\mu \star i)$ is undefined.

We identify the following three properties of Δ:

(1) $(\Theta^*)^*(A) = \Theta(A)$: When Θ^* does not query its oracle, $(\Theta^*)^*$ behaves just as Θ^*, and this is identical to the behavior of $\Theta(A)$. When Θ^* does query an oracle, $\mu \star i$ from a Θ query is replaced by a query at $\mu \star (1-i)$. $(\Theta^*)^*$ will be required to make the same form of replacement, so $(\Theta^*)^*$ will query its oracle at the string $\mu \star (1-(1-i)) = \mu \star i$, exactly as Θ will in its computation.

(2) Under the map on degrees induced by Δ, $deg(A_1)$ is the image of $deg(A_0)$: Consider the linear-time reduction $\Theta(A) = A_0$ given by $\Theta(A)(\sigma) = 0$ if σ ends in 1, and $\Theta(A)(\sigma) = A(\sigma)$ if σ ends in 0. Then it is easy to see that $\Theta^*(A) \equiv_p A_1$, since $\sigma \in A_1 \iff \sigma = \mu \star 1$ and $\mu \star 0 \in \Theta^*(A)$.

(3) Θ^* is a polynomial-time bounded Turing reduction, and the degree of Θ^*'s time bound is the same as that of θ.

2.3.1. Building the generic set

Our definition of genericity will be tied to the definition of ultra-sparseness. Fix a linear-time computable set of strings S satisfying conditions (1), (2) and (3) in the definition of ultra-sparseness.

Since the composition of two *PTIME* reductions is again a *PTIME* reduction, we phrase our construction in terms of satisfying requirements of the form $\Theta(A) \neq \Psi(A)$, rather than in terms of $\Theta(A) \neq \Phi(\Psi(A))$.

For each pair of *PTIME* reductions Θ and Ψ, we want to force the following statement to be true:

If there exist infinitely many σ such that there exists a condition q such that

(1) $q \supseteq p_{A,|\sigma|}$,

(2) for any $\tau \in \text{domain}(q) - \text{domain}(p_A(|\sigma|)$, $q(\tau) = 1$ implies $\tau \in S$,

(3) $|q| \leqslant \theta(|\sigma|), \psi(|\sigma|)$,

(4) There is at most one $\tau \in S$ in $\text{domain}(q) - \text{domain}(p_A(|\sigma|) -$ and if such a τ exists, then it is the least element of S not in $\text{domain}(p_{A,|\sigma|})$, and

(5) $\Theta(q)(\sigma) \neq \Psi(q)(\sigma)$,

then there is a σ such that $\Theta(A)(\sigma) \neq \Psi(A)(\sigma)$.

We call a set which satisfies this statement for all *PTIME* reductions Θ and Ψ a *generic* set. The set A which we construct will be generic.

The technique we use for putting the construction together is a device used by Shinoda and Slaman [4] in their constructions for interpreting arithmetic in $\mathscr{R}$. Their technique is motivated and described in detail there; we give only a brief sketch here.

At stage $s+1$ in the construction we decide the values of $A(\mu \star 0)$ and $A(\mu \star 1)$, where μ is the $(s+1)$st string in S. (The construction also enumerates S.) The search for strings σ at which to diagonalize is bounded by the time it takes to compute the default values of $A(\mu \star 0)$ and $A(\mu \star 1)$ if no strategy acts to diagonalize at stage $s+1$.

We introduce some more notation. Let p_A^s denote the condition which contains the information about A which is determined by stage s. This will include $A(\eta \star i)$ for $i = 0, 1$ and $\eta \in S$ such that η is the tth element of S and $t \leqslant s$. We let $R(\Theta, \Psi)$ denote the following requirement: $\exists$ infinitely many $\sigma(\exists q \supseteq p_{A,|\sigma|}, q$ is consistent with ultra-sparseness and

$$\Theta(q)(\sigma) \neq \Psi(q)(\sigma)) \Longrightarrow \exists \sigma(\Theta(A)(\sigma) \neq \Psi(A)(\sigma))$$

Fix a priority ordering of these requirements.

We use $\mathscr{S}_i$ to denote the following strategy for satisfying the requirement $R(\Theta, \Psi)$ of priority i:

At stage s in the construction, if we *see* an extension q of p_A^s and a string σ such that $\Theta(q)(\sigma) \neq \Psi(q)(\sigma)$, then make A extend q (and say that the strategy has acted).

Next we must define what it means to *see* an extension q and a string σ. If we explicitly bound the search for σ, then the search for q is implicitly bounded by $max\{\theta(|\sigma|), \psi(|\sigma|)\}$. Let $out(s)$ be the number of steps it takes to run the construction through stage s. (Note that in general $out(s)$ will be much larger than s.) Let $\mathscr{S}_0, \mathscr{S}_1, ..., \mathscr{S}_s$ be the strategies corresponding to the $s+1$ highest priority requirements. These will be the only strategies allowed to act at stage $s+1$.

Our action at stage $s+1$ will be to define $A(\eta_{s+1} \star 0)$ and $A(\eta_{s+1} \star 1)$, where η_{s+1} is the $(s+1)$st string in S. We will define a function, $in_i(s+1)$, which gives a bound on the length of strings to be considered by the strategy $\mathscr{S}_i$ at stage $s+1$. The construction is arranged so that if no strategy of priority higher than $\mathscr{S}_i$ acts at stage $s+1$, then $in_i(s+1)$ is large enough to compute $A(\eta_{s+1} \star 0)$ and $A(\eta_{s+1} \star 1)$. We use $in_i(s+1)$ as the bound on the length of strings strategy $\mathscr{S}_i$ is allowed to consider during stage $s+1$. Intuitively, $in_i(s+1)$ is the number of steps it takes to run the construction up to stage s and to determine all activity of priority lower than i. This is equal to the

number of steps it takes to compute the values of A at the new decision points if no strategy of priority higher than i acts.

Formally,

$$in_s(s+1) = max(out(s), |\eta_{s+1}| + 1)$$
$$in_i(s+1) = 2^m,$$

where $m = max\{\theta(in_{i+1}(s+1)), \psi(in_{i+1}(s+1)), in_{i+1}(s+1)\}$, where θ and ψ are the polynomial-time bounds involved in strategy $\mathscr{S}_{i+1}$.

Thus, $in_i(s+1)$ is the time it takes to run strategy $\mathscr{S}_{i+1}$ on its input possibilities, which are strings of length $\leqslant in_{i+1}(s+1)$. So $in_0(s+1)$ is the time it takes to run the construction through stage $s+1$.

A construction which acts according to the above interleaving of strategies will produce a generic set. We give a brief verification of this below.

We need to establish that all of the requirements $R(\Theta, \Psi)$ are satisfied. If they are all not satisfied, consider the highest priority requirement which is not satisfied, call it $R(\Theta, \Psi)$. Let i be the priority of $R(\Theta, \Psi)$.

It must be the case that $\mathscr{S}_i$ never acted, otherwise $R(\Theta, \Psi)$ would be satisfied. Let s_0 be a stage beyond which no requirement of priority higher than i acts. Choose σ and $s > s_0$ such that $s > i$ and $p_{A,|\sigma|} = p_A^s$ and $\exists q \supseteq p_{A,|\sigma|}(\Theta(q)(\sigma) \neq \Psi(A)(\sigma))$. We need to show that $|\sigma| \leqslant in_i(s+1)$, since if this is the case then $\mathscr{S}_i$ will act at stage $s+1$ to satisfy $R(\Theta, \Psi)$.

Since $R(\Theta, \Psi)$ does not act at stage $s+1$, and no requirement of priority higher than i acts at stage $s+1$, $in_i(s+1)$ must be large enough to compute the values of $A(\eta \star 0)$ and $A(\eta \star 1)$. But $|\eta| > |p_{A,|\sigma|}| = |p_A^s|$. (This is true because η is a *new* decision point at stage $s+1$.) Thus, $|\sigma|$ is not large enough to compute $A(\eta \star 0)$ and $A(\eta \star 1)$, but $in_i(s+1)$ is large enough to compute $A(\eta \star 0)$ and $A(\eta \star 1)$. Hence $in_i(s+1) > |\sigma|$, and so $R(\Theta, \Psi)$ would have acted at stage $s+1$ in the construction, a contradiction.

3. Formal construction of A

A is constructed in stages by a recursive construction. During the construction we also enumerate a linear-time computable set of strings S, which will witness the ultra-sparseness of A. At stage s we define p_A^s and S_s. Then

$$S = \bigcup_{s \in \omega} S_s \quad \text{and} \quad A = \bigcup_{s \in \omega} p_A^s$$

Construction

Stage 0: $p_A^0(\emptyset) = 0$, p_A^0 is undefined elsewhere. $S_0 = \emptyset$.

Stage $s+1$: Compute $out(s)$. (This is equal to the number of steps it takes to run the first s steps of the construction.) Let η be the least string such that $|\eta| > out(s)$

and $p_A(\eta \star i)$ is undefined for $i = 0, 1$, and $|\eta| > \varphi(|\tau|)$ for all τ such that $|\tau| \leqslant$ the length of the longest string in S_s and all polynomials φ mentioned in requirements of priority higher than or equal to $s+1$. Let $S_{s+1} = S_s \cup \{\eta\}$.

We say that the requirement of priority i, call it $R(\Theta, \Psi)$, requires attention at stage $s+1$ if $i \leqslant s+1$ and

$$\exists \sigma\ (|\sigma| \leqslant in_i(s+1) \text{ and } \exists q \supseteq p_A^s$$
$$(|q| \leqslant \max(\theta(|\sigma|), \psi(|\sigma|))) \text{ and}$$
$$(q(\tau) = 1 \Rightarrow p_A^s = 1 \text{ or } \tau = \eta \star 0 \text{ or } \tau = \eta \star 1) \text{ and } \Theta(q)(\sigma) \neq \Psi(q)(\sigma))$$

Let $R(\Theta, \Psi)$ be the highest priority requirement which requires attention at stage $s+1$, and let q be the corresponding condition.

Define $p_A^{s+1} = q$. If there is no such $R(\Theta, \Psi)$, define $p_A^{s+1}(\tau) = p_A^s(\tau)$ for all τ in the domain of p_A^s, and $p_A^{s+1}(\tau) = 0$ for all other τ such that $|\tau| \leqslant |\eta| + 1$.

End of construction.

4. Verification that the construction succeeds

Lemma 4.1. *The set constructed in Section 3 is generic, i.e. it meets all of the requirements* $R(\Theta, \Psi)$.

Proof. The strategy for each $R(\Theta, \Psi)$ acts at most once and can never be injured after it acts. Thus, the discussion at the end of Section 2 (showing that each requirement has a chance to act) actually shows that each requirement is satisfied. □

Lemma 4.2. $A_0 \not\leqslant_{pT} A_1$ *and* $A_1 \not\leqslant_{pT} A_0$.

Proof. Suppose $A_0 = \Phi(A_1)$ for some *PTIME* reduction Φ. Take Θ and Ψ, *PTIME* reductions such that $\Theta(A) = A_0$, $\Psi(A) = A_1$, Θ queries its oracle only on strings ending in 0, and Ψ queries its oracle only on strings ending in 1. Since these values are completely independent, there will be infinitely many strings $\sigma \in S$ and infinitely many stages s such that

$$\exists q \supseteq p_A^s(\Theta(q)(\sigma \star 0) \neq \Phi(\Psi(q))(\sigma \star 0) \text{ and } q \text{ is consistent with } S).$$

Since A is generic, we can conclude that $\Theta(A) \neq \Phi(\Psi(A))$, and so $A_0 \neq \Phi(A_1)$. Similarly, $A_1 \not\leqslant_{pT} A_0$. □

Consider the map f on $\mathscr{P}^A$ induced by Δ. We verify that f is an automorphism of $\mathscr{P}^A$. The preceding lemma guarantees that the automorphism is nontrivial.

Without loss of generality (by Proposition 2), we consider only *PTIME* reductions Θ such that for any string σ, there is at most one string η such that the computation

$\Theta(A)(\sigma)$ queries "$\eta \star i \in A$?" for $i = 0, 1$, where $A(\eta \star i)$ is not determined by $p_{A,|\sigma|}$ and $\eta \in S$.

Lemma 4.3. $\Theta(A) \leqslant_{pT} \Psi(A) \Longrightarrow \Theta^*(A) \leqslant_{pT} \Psi^*(A)$.

Proof. Suppose that $\Phi(\Psi(A)) = \Theta(A)$. We define a *PTIME* reduction, $\widehat{\Phi}$ so that $\widehat{\Phi}(\Psi^*(A)) = \Theta^*(A)$. In order to define $\widehat{\Phi}$, we describe the computation of $\widehat{\Phi}(X)(\sigma)$, where X is an arbitrary oracle set and σ is a string.

To compute $\widehat{\Phi}(X)(\sigma)$:

Let $l = \max(\psi(\varphi(|\sigma|), \theta(|\sigma|), |\sigma|)$. Compute $p_{A,l}$.

Case 1. If $p_{A,l}$ determines $\Theta^*(A)(\sigma)$ unambiguously then answer $\Theta^*(A)(\sigma)$.

Case 2. Otherwise, whenever Φ queries its oracle with "$\tau \in \Psi(A)$?", if $p_{A,l}$ determines $\Psi(A)(\tau)$ unambiguously, then answer $\Psi(A)(\tau)$.

($\diamondsuit$) Otherwise answer $X(\tau)$.

Notice that the only oracle queries in this computation occur at ($\diamondsuit$).

Claim 4.4. *For all but finitely many* σ, $\widehat{\Phi}(\Psi^*(A))(\sigma) = \Theta^*(A)(\sigma)$.

We will show that if σ is such that $\widehat{\Phi}(\Psi^*(A))(\sigma) \neq \Theta^*(A)(\sigma)$ then there is a q such that $\Phi(\Psi(q))(\sigma) \neq \Theta(q)(\sigma)$. Hence, if there were infinitely many σ satisfying $\widehat{\Phi}(\Psi^*(A))(\sigma) \neq \Theta^*(A)(\sigma)$, then, by Lemma 4.1 and the requirements, it would follow that $\Phi(\Psi(A)) \neq \Theta(A)$, contrary to the hypothesis.

We intend to show that if σ is such that

$$\widehat{\Phi}(\Psi^*(A))(\sigma) \neq \Theta^*(A)(\sigma),$$

then there is a q such that $\Phi(\Psi(q))(\sigma) \neq \Theta(q)(\sigma)$. Hence, if there were infinitely many σ satisfying

$$\widehat{\Phi}(\Psi^*(A))(\sigma) \neq \Theta^*(A)(\sigma),$$

then, by Lemma 4.1 and the requirements, it would follow that $\Phi(\Psi(A) \neq \Theta(A)$, contrary to the hypothesis.

Fix σ and suppose $\widehat{\Phi}(\Psi^*(A))(\sigma) \neq \Theta^*(A)(\sigma)$. Then $\widehat{\Phi}(\Psi^*(A))(\sigma)$ is not computed via case 1 of the definition of $\widehat{\Phi}$. Consider the condition $q \supseteq p_{A,|\sigma|}$ defined as follows. Let η be the unique string such that $\eta \in S$ and the computation $\Theta(A)(\sigma)$ queries its oracle at $\eta \star 0$ or $\eta \star 1$. Define $q(\eta \star i) = A(\eta \star (1 - i))$ for $i = 0, 1$, $q(\tau) = p_{A,l}(\tau)$ for all τ such that $p_{A,l}(\tau)$ is defined, and $q(\tau) = 0$ for all other τ of length less than l.

Subclaim a. $\Theta(q)(\sigma) = \Theta^*(A)(\sigma)$.

Since $\widehat{\Phi}(\Psi(A))(\sigma)$ is not computed via case 1 in the definition of $\widehat{\Phi}$, there is a string $\eta \in S$, and an $i = 0$, or 1 such that $|\eta| + 1 < \theta(|\sigma|) \leqslant l$ and $A(\eta \star i)$ is queried in the computation $\Theta(A)(\sigma)$ and $A(\eta \star i)$ is not determined by $p_{A,l}$. We claim that there

is no string $\gamma \neq \eta$ and $\gamma \in S$ and no $i = 0, 1$ such that $A(\gamma \star i)$ is queried during the computation of $\Theta(A)(\sigma)$ and $A(\gamma \star i)$ is determined by $p_{A,l}$ but $A(\gamma \star i)$ is not determined by $p_{A,|\sigma|}$. This is the case since there is at most one string γ in S such that $A(\gamma \star i)$ is queried during the computation $\Theta(A)(\sigma)$, and there is such a string which is not determined by $p_{A,l}$. Thus, the relevant points in the computations $\Theta(A)(\sigma)$ and $\Theta^*(A)$ are $\eta \star 0$ and $\eta \star 1$, and so the subclaim is true by the definition of Θ^*.

Subclaim b. $\Phi(\Psi(q))(\sigma) = \widehat{\Phi}(\Psi^*(A))(\sigma)$.

It suffices to see that all oracle queries in the computations $\Phi(\Psi(q))(\sigma)$ and $\widehat{\Phi}(\Psi^*(A))(\sigma)$ are answered identically. Consider an oracle query "$\tau \in X$?". If τ is such that $\Psi(A)(\tau)$ is determined by $p_{A,l}$, then the answers to the query in both the Φ and $\widehat{\Phi}$ computations are $\Psi(A)(\tau)$. (This is guaranteed by case 2 in the definition of $\widehat{\Phi}$.)

Note that there can be at most one string $\eta \in S$ such that for $i = 0, 1$, $l \geqslant |\eta \star i|$ and $A(\eta \star i)$ is not determined by $p_{A,l}$. To see this, suppose that there were two such strings, η_1 and $\eta_2 \in S$ with $|\eta_1| < |\eta_2|$. Then $|\eta_2|$ is large enough to compute $A(\eta_1 \star i)$ (by the definition of ultra-sparse) and $l > |\eta_2|$, so $p_{A,l}$ determines $A(\eta_1 \star i)$, contrary to our assumption.

For queries "$\tau \in X$?" where $\Psi(A)(\tau)$ is not determined by $p_{A,l}$, we must see that $\Psi^*(A)(\tau) = \Psi(q)(\tau)$, i.e. that $\Psi^*(A)(\tau)$ is computed by applying Ψ to q. By definition, $\Psi^*(A)(\tau)$ is computed by applying Ψ to $p_{A,|\tau|} \star \alpha$, where α is the finite partial function on strings constructed by switching the roles of $\eta \star 0$ and $\eta \star 1$, the two relevant decision points in the computation. We know that $|\tau| \leqslant \varphi(|\sigma|)$, so $|\tau| \leqslant l$, so $p_{A,|\tau|}$ is extended by $p_{A,l}$. $\Psi(A)(\tau)$ is not determined by $p_{A,l}$, but it is determined by $A \upharpoonright l$, so the two relevant points in the computation must be $\eta \star 0$ and $\eta \star 1$. Thus, we see that $q \supseteq p_{A,|\tau|} \star \alpha$, and so $\Psi(A)(\tau) = \Psi^*(q)(\tau)$. □

Lemmas 4.1–4.3 show that f induces a nontrivial automorphism of $\mathscr{R}(\leqslant deg(A))$ and of $\mathscr{P}^A$, and so Theorems 1 and 1′ are proved.

5. Discussion of Theorems 2 and 3

We recall Theorem 2.

Theorem 2. (1) *There is a recursive oracle A and a nontrivial automorphism of $\mathscr{P}^A$, induced by a map f on $PTIME^A$ such that for all natural numbers $k \geqslant 1$, $f(DTIME^A(n^k)) = DTIME^A(n^k)$.*

(2) *For any natural numbers $k \neq j \geqslant 1$ there is a recursive oracle, A, and a nontrivial automorphism of $\mathscr{P}^A$, induced by a map f on $PTIME^A$ such that for some $B \in DTIME^A(n^k)$, $f(B) \in DTIME^A(n^j)$, and for all $l < k$, $B \notin DTIME^A(n^l)$, and for all $m < j$, $f(B) \notin DTIME^A(n^m)$.*

Proof. (1) follows from the observation that if $k \geqslant 1$ and Θ is a *PTIME* reduction with time bound of order n^k, then Θ^* is also a *PTIME* reduction with time bound of order n^k. The Θ^* computation takes time equal to the time of the Θ computation plus a linear amount of time to do the simulation of A.

(2) Suppose that we want to move $B \in DTIME^A(n^k)$ to $C \in DTIME^A(n^j)$. We know that our construction moves A_0 to A_1, so we pad the coding of A_0 and A_1 into A so that $A_0 \in (DTIME^A(n^k) - DTIME^A(n^{k-1}))$ and $A_1 \in (DTIME^A(n^j) - DTIME^A(n^{j-1}))$. The modified coding is then propagated through the proof of Theorem 1 (for instance it induces a modified definition of ultra-sparse). We use the following modified coding of A_0 and A_1 into A.

Let u_k and u_j be polynomials such that for all n, $u_k(n) \neq u_j(n)$ and u_k has degree k and u_j has degree j:

$$\sigma \in A_0 \Longleftrightarrow \sigma \star 0^{u_k(|\sigma|)} \in A$$
$$\sigma \in A_1 \Longleftrightarrow \sigma \star 0^{u_j(|\sigma|)} \in A$$

The definition of ultra-sparse is modified so that the only strings in A are now of the form $\sigma \star 0^{u_k(|\sigma|)}$ or $\sigma \star 0^{u_j(|\sigma|)}$, where $\sigma \in S$. It remains to be seen that A_0 and A_1 are not in any $DTIME^A(n^l)$ for $l < k$ and $l < j$, respectively. This will be guaranteed by the genericity of A. For suppose that, say, $A_0 = \Delta(A)$, where Δ is a *PTIME* reduction with time bound δ, δ is a polynomial of degree $< k$. Then $\delta(n) < u_k(n)$ for cofinitely many n. Then for $\sigma \in S$ with $\delta(|\sigma|) < u_k(|\sigma|), A \upharpoonright \delta(|\sigma|)$ determines the value of $\Delta(A)(\sigma)$, but $A(\sigma \star 0^{u_k(|\sigma|)})$ determines the value of $A_0(\sigma)$, and $A(\sigma \star 0^{u_k(|\sigma|)})$ is not determined by $A \upharpoonright \delta(|\sigma|)$. Thus, there is ample opportunity to diagonalize and make $A_0 \neq \Delta(A)$. Since A is generic, it will be the case that $A_0 \neq \Delta(A)$. The argument for A_1 is symmetric. □

We turn now to Theorem 3.

Theorem 3. *There is a recursive oracle A and a map $g : PTIME^A \longrightarrow PTIME^A$ such that g is an automorphism of $\mathscr{L}_A$ and g induces an automorphism of $\mathscr{P}^A$.*

Proof. We first produce an automorphism of $\mathscr{L}_A^*$ which simultaneously induces an automorphism of $\mathscr{P}_A$. We want to establish the following additional requirements:

$$\Theta(A) \subseteq^* \Psi(A) \Longrightarrow \Theta^*(A) \subseteq^* \Psi^*(A)$$

So, in the course of the construction we try to force $\Theta(A) \not\subseteq^* \Psi(A)$ whenever possible; i.e. there are infinitely many σ such that $\sigma \in \Theta(A)$ and $\sigma \notin \Psi(A)$.

We rewrite these requirements as finitary requirements and then strengthen our definition of generic to encompass them. The finitary requirements are: $Q(\Theta, \Psi, n)$: If there exist at least $n-1$ strings σ such that $\sigma \in \Theta(A)$ and $\sigma \notin \Psi(A)$ and there exist infinitely many σ such that $\exists q \supseteq p_{A,|\sigma|}$, q is consistent with S and $\sigma \in \Theta(q)$ and $\sigma \notin \Psi(q)$, then there exist at least n strings σ such that $\sigma \in \Theta(A)$ and $\sigma \notin \Psi(A)$.

Then our original construction, with these additional requirements thrown in, will produce a map $f : PTIME^A \longrightarrow PTIME^A$ such that for $B, C \in PTIME^A$

$$B \subseteq^* C \iff f(B) \subseteq^* f(C)$$
$$B \leqslant_{pT} C \iff f(B) \leqslant - pf(C)$$

Now, exactly as in [5, p. 342] it is possible to show that there is a permutation of $\{0,1\}^*$, g, such that $g(B) =^* f(B)$. g gives an automorphism of $\mathscr{L}_A$, since $B \subseteq C \Rightarrow g(B) \subseteq g(C)$, and g induces an automorphism of $\mathscr{P}_A$ since for $B, C \in PTIME^A, B \leqslant_{pT} C \Rightarrow f(B) \leqslant_{pT} f(C)$, but $f(B) =^* g(B)$, so $f(B) \equiv_p g(B)$ and likewise $f(C) \equiv_p g(C)$, so $g(B) \leqslant_{pT} g(C)$. □

References

[1] K. Ambos-Spies, Inhomogeneities in the polynomial-time degrees: the degrees of super sparse sets, Inform. Processing Lett. 22 (1986) 113–117.
[2] S.A. Cook, The complexity of theorem proving procedures, in: Proc. 3rd Ann. ACM Symp. on Theory of Comp. (1971) 151–158.
[3] R. Ladner, On the structure of polynomial-time reducibility, J. Assoc. Comput. Machinery 22 (1975) 155–171.
[4] J. Shinoda and T.A. Slaman, On the theory of the *PTIME* degrees of recursive sets, to appear.
[5] R.I. Soare, Recursively Enumerable Sets and Degrees (Springer, Berlin 1987).