Reprinted from JOURNAL OF COMPUTER AND SYSTEM SCIENCES
All Rights Reserved by Academic Press, New York and London

Vol. 45, No. 3, December 1992
Printed in Belgium

Nondiamond Theorems for Polynomial Time Reducibility

ROD DOWNEY*

Mathematics Department, Victoria University,
PO Box 600, Wellington, New Zealand

We investigate the structure of recursive sets under polynomial time Turing reducibility. In particular, we solve a question of Ambos-Spies by constructing a polynomial time degree that is not the supremum of a minimal pair. The proof is of some technical interest as it uses π_2 priority arguments and the speedup phenomenon. © 1992 Academic Press, Inc.

1. INTRODUCTION

The notions of polynomial time Turing (p-T) and polynomial time many–one (p-m) reducibilities were introduced by Cook [7] and Karp [10], respectively. These reducibilities give natural measures of the relative difficulties of combinatorial problems. The equivalence classes consist of problems of the same level of complexity and are called *degrees*. Both the p-T and p-m degrees under the induced orderings form natural upper semilattices (usl's) which we will denote by $\langle \mathbf{R}_T^p, \leqslant \rangle$ and $\langle \mathbf{R}_m^p, \leqslant \rangle$, respectively.

The analogous structures for classical reducibilities have been studied for many years (see e.g. Rogers [17], Soare [22], or Odifreddi [16]). Both $\mathbf{R}_T^p$ and $\mathbf{R}_m^p$ have least element $\mathbf{0}$ the degree of the *p*-time sets. The study of structural properties of $\mathbf{R}_T^p$ and $\mathbf{R}_m^p$ was initiated by Ladner [11, 12]. In [11, 12] Ladner showed that both $\mathbf{R}_T^p$ and $\mathbf{R}_m^p$ are countable dense usl's that are not lattices and a number of other interesting results.

In this paper we are concerned with one of Ladner's results. That is, there exist *minimal pairs* of degrees in $\mathbf{R}_m^p$ and $\mathbf{R}_T^p$: there exist $\mathbf{a}, \mathbf{b} \neq \mathbf{0}$ with $\mathbf{a} \cap \mathbf{b} = \mathbf{0}$, where $\cap$ denotes the partial infimum operator. Intuitively, if $A \in \mathbf{a}$ and $B \in \mathbf{b}$ then although both A and B cannot be computed in p-time, the information they code is so different that the only sets they *both* can compute in p-time are the p-time sets themselves. Another way of saying that minimal pairs exist is to say that the diamond lattice embeds into (e.g.) $\mathbf{R}_T^p$, preserving $\mathbf{0}$ (via $\mathbf{0}, \mathbf{a}, \mathbf{b}, \mathbf{a} \cup \mathbf{b}$).

* Research partially supported by a US/NZ binational grant. The author wishes to thank Klaus Ambos-Spies and Steve Homer for helpful conversations concerning this topic, and thank them for their hospitality during the author's visit to Heidelberg, where they introduced the author to the problem addressed here.

0022-0000/92 $5.00

Ladner's paper introduced a fundamental technique called *delayed diagonalization.* This technique was later refined by Briedbart [5], Landweber *et al.* [13]. Melhorn [14, 15], Chew and Machtey [6] and others.

In particular, Landwever *et al.* [13] and Chew and Machtey [6] extended Ladner's minimal pair technique to show that each non-zero p-T (or p-m) degree bounds a minimal pair, and elegant structural variations of such arguments can be found in the work of Schöning [18, 19], Borodin *et al.* [4], and others.

A natural question raised by all of this work was addressed by Ambos-Spies [1, 2]: Is every p-time degree the top of a diamond? Ambos-Spies answered this question for $\mathbf{R}_m^p$ by showing that there exists $\mathbf{a} \neq \mathbf{0}$ such that for all $\mathbf{0} < \mathbf{c}, \mathbf{d} < \mathbf{a}$ if $\mathbf{c} \cap \mathbf{d} = \mathbf{0}$ then $\mathbf{c} \cup \mathbf{d} \neq \mathbf{a}$. That is, $\mathbf{a}$ is not the top of a diamond in $\mathbf{R}_m^p$. This work was extended considerably by Ambos–Spies *et al.* [3], who showed that any elementary recursive set which is hard for deterministic exponential time cannot be supremum of a minimal pair in $\mathbf{R}_m^p$ and, further, the question of whether NP complete sets are tops of minimal pairs (in $\mathbf{R}_m^p$) is oracle dependent.

Each of these results used techniques that did not seem to pertain to $\mathbf{R}_T^p$ and the question of whether such a "nondiamond" theorem held for $\mathbf{R}_T^p$ remained open. This question was first explicitly raised in Ambos-Spies [1]. Our result is to solve Ambos-Spies question affirmatively.

1.1. THEOREM. *In* $\mathbf{R}_T^p$ *there exists* $\mathbf{a} \neq \mathbf{0}$ *such that* $\mathbf{a}$ *is not the top of a diamond.*

Intuitively, Theorem 1.1 says that if $A \in \mathbf{a}$ the information in A cannot be partitioned into two sets so that the information in the sets is so different as to form a minimal pair.

We remark that 1.1 is also interesting from a technical point of view since its proof technique uses a π_2 priority argument. Early results for $\langle \mathbf{R}_T^p, \leqslant \rangle$ all used variations on delayed diagonalisation. The first use of other methods to preserve infima was the use of a finite injury (π_1) priority argument by Ambos-Spies [1, 2] to solve a question of Mehlhorn [14, 15] and Schöning [19]: Ambos-Spies showed that every $\mathbf{a} \neq \mathbf{0}$ is half of a minimal pair [1, 2]. We remark that the Ambos-Spies–Homer–Soare techniques depend heavily on the distributive of $\mathbf{R}_m^p$ and seem inappropriate for $\mathbf{R}_T^p$.

Extensions of this technique can also be found in Shinoda and Slaman [20] and Shore and Slaman [21]. Crucial to these arguments (and ours) is the use of a sort of *speedup strategy* as part of the construction.

Our construction will be a π_2 argument which means that it requires a π_2-complete oracle to figure out how we have met our requirements. Our argument could be described as the p-time analogue of the gap-co-gap $\mathbf{0}'''$ method from classical recursion theory and is more complex than either Ambos-Spies [1, 2] or Shore and Slaman [21], since we need to meet noninfimum requirements rather than infimum ones. (These latter results need π_1 arguments.)

The only other place where a π_2 argument has been used is in Shinoda and Slaman [20]. It is also hoped that our argument will make the proof of Shinoda

and Slaman [20] rather more accessible. In particular, our strategies do not use the language of forcing used in [20, 21]. We make further comments regarding this paper and the Shinoda and Slaman paper following the discussion of the procedure $R(e, i, s, t)$ in the basic module in the proof of 1.1.

The degree **a** that we construct in 1.1 is quite complicated (not elementary) and this seems a drawback of the speedup technique. That is, it does not seem to be able to construct sets in natural classes such as NP or EXPTIME, or other elementary classes.

2. Preliminaries

Sets will be identified with their characteristic functions, and we do not distinguish between a machine and the language accepted by a machine. We suppose that all time bounded machines halt on all inputs. We will denote polynomial time bounded oracle Turing machines by uppercase Greek letter ($\Phi, \Gamma, \Psi, \Delta, \ldots$) and the corresponding lower case letter ($\varphi, \gamma, \psi, \delta, \ldots$) will denote the relevant "use" function, that is, the polynomial bound on the computation. Thus $\Phi(A; x)$ halts in $<\varphi(|x|)$ steps. φ will always be monotone. We will let $\Psi_{e,s}(A_s; x)$ denote s steps in the enumeration of the eth p-time machine on input x with oracle A_s in some simultaneous p-time enumeration of all p-time machines.

As usual our alphabet will be $\Sigma = \{0, 1\}$ unless otherwise stated, and so sets (or languages) are subsets of Σ^*. We will let ω denote the natural numbers, we will let $A \oplus B$ denote $\{0x: x \in A\} \cup \{1x: x \in B\}$ and let $A[x] = \{z: z \in A \text{ and } |z| \leqslant |x|\}$.

3. The Main Result

In this section we prove 1.1 which we restate for convenience below.

3.1. Theorem. *In* $\mathbf{R}_{\mathrm{T}}^{\mathrm{p}}$ *there exists* $\mathbf{a} > \mathbf{0}$ *such that for all* $\mathbf{c}, \mathbf{d} > \mathbf{0}$ *if* $\mathbf{c} \cup \mathbf{d} = \mathbf{a}$ *then* $\mathbf{c} \cap \mathbf{d} \neq \mathbf{0}$.

To prove 1.1 we shall build a recursive set A together with auxiliary recursive sets $\{Q_e: e \in \omega\}$ to satisfy the requirements,

$$P_e: A \neq \Psi_e(\varnothing)$$

$$R_e: \Gamma_e(\Phi_e(A) \oplus \Delta_e(A)) = A \rightarrow (\Phi_e(A) \in P \text{ or } \Delta_e(A) \in P \text{ or } \forall i(R_{e,i})),$$

where

$$R_{e,i}: Q_e \leqslant_{\mathrm{T}}^{\mathrm{p}} \Phi_e(A), \Delta_e(A) \quad \text{and} \quad Q_e \neq \Psi_i(\varnothing).$$

Here $\{\Psi_e : e \in \omega\}$ is an enumeration of all p-time machines and $\langle \Gamma_e, \Phi_e, \Delta_e \rangle_{e \in \omega}$ is an enumeration of all triples of such machines.

Before we give the details of the construction we will focus on the strategies we will use to meet the requirements above. We will build $A \subset \{0^n : n \in \omega\}$.

Initially we examine the easiest requirements, the P_e. To fit in with the other requirements these are satisfied in a slightly different way from the usual diagonalization argument. We will use the procedure $P(e, x, s)$ described below.

PROCEDURE $P(e, x, s)$. To meet P_e we will take on elements $x = 0^n$ for some n at stage s (called a *follower* of P_e) and await a stage $t > s$ such that $\Phi_{e,t}(\varnothing; x)\downarrow$. At stage t we will then enumerate x into A_t iff $\Phi_{e,t}(\varnothing; x) = 0$, and protect x from withdrawal (by other requirements) with priority e. The output of this procedure will be $A_t[x]$ and t.

We now turn to the discussion of the satisfaction of the R_e requirements. To aid the discussion in the procedures to follow we will use the phrase "*initialize*." By this we mean that if a requirement R initializes all requirements R' of lower priority at stage s, then it will *assert control* over $A[s]$, thereafter making $A[s]$ forbidden to such R'. In particular, if R' is $R_{e,i}$ for some e, i then in the procedure $R(e, i, t_1, t_2)$ below we reset $t_1 = s$. The reader may wish to think of this as a restraint.

Before we deal with the "α-module" (i.e., the coherence of the strategies) we will discuss the *basic module* for $R_{e,i}$ below. Dropping the "e" we must make either $\Gamma(\Phi(A) \oplus \Delta(A)) \neq A$, $\Phi(A) \in P$ or $\Delta(A) \in P$ or build $Q \leqslant_{\mathrm{T}}^{\mathrm{p}} Q(A)$, $\Delta(A)$ to make $Q \neq \Psi_i(\varnothing)$.

We will need the auxiliary function $\rho(|x|) = \max\{\varphi(\gamma(|x|)), \delta(\gamma(|x|))\}$.

To meet $R_{e,i}$ we will use Procedure $R(e, i, s, t)$. When we enter this procedure (at stage t), $R_{e,i}$ will assert control over those z with $s < |z| \leqslant t$, but will have no control over $A[s]$ (presumably this is restrained by higher priority requirements).

Entering the procedure there will either be a single $R_{e,i}$-controllable A-number $z_1 = 0^n$ for some n (with $s < n \leqslant t$) or there will be two such A-numbers. If there are two such A-numbers, z_1, z_2, then $z_2 = 0^m$, $z_1 = 0^n$, and without loss of generality $n < m$.

Being an A-number means that it may possibly enter A. If there are two A-numbers over which $R_{e,i}$ can assert control, then $R_{e,i}$ will be either in *state* Φ or *state* Δ. As we will see, intuitively, if $R_{e,i}$ is in state Φ it currently *appears that* $\Delta(A) \in P$. At the end of procedure $R(e, i, s, t)$ we will either win $R_{e,i}$ and initialize all R' of lower priority than $R_{e,i}$, or $R_{e,i}$ will still be in the same state as it was when we entered. In this latter case $R_{e,i}$, will release control (forever) of z_1 but there will be a new A-number z_3 with $|z| > |z_2|$ over which $R_{e,i}$ will assert control. Finally, the reader should note that $A(q) = 0$ for all q which are not A-numbers.

PROCEDURE $R(e, i, s, t)$.

Remark. This procedure has the following parameters and variables:

$A^*(m, n, \varepsilon x, \varepsilon y)$—this is an incarnation of A_m. Such sets will be examined to see if they produce changes to obtain disagreements.

s. A lower bound upon which we have fixed A.

t. We are concerned with k such that $s < k \leqslant t$.

t_1. The stage we need to wait until to make the t-computations visible.

u. A parameter for the Ψ_i-computations.

t_3, t_4. These leave the same roles as t and t_1 in parts 3 of the procedure.

z_1, z_2. Followers targeted for A (these are numbers used to induce changes).

q. Numbers targeted for Q.

x_1, x_2. Numbers that A changes (via z_1 and z_2) are reflected in via Φ and Δ.

The procedure consists of the following steps:

1. If $R_{e,i}$ can assert control over two A-numbers (at stage t) z_1 and z_2 go to 3. If not, $R_{e,i}$ will be able to assert control over one A-number z_1. We wait till stage $t_1 = 2^{\rho(t)+1}$. Note that we can now "see" all of the (Γ, Φ, Δ)-computations on A for $|z| \leqslant t$.

2. Define $A^*(m, n, \varepsilon x, \varepsilon y)$, where $\varepsilon x = x$ or $\neg x$ and $\varepsilon y = y$ or $\neg y$ as the result of setting at stage m,

$$\begin{aligned} A_m(z) = A_s(z) \quad & \text{for} \quad |z| \leqslant s \\ = 0 \quad & \text{if} \quad z \neq x, y \quad \text{and} \quad |z| < n \\ = 0 \quad & \text{if} \quad z = x \quad \text{and} \quad \varepsilon x = \neg x \\ & \text{or} \quad z = y \quad \text{and} \quad \varepsilon y = \neg y \\ = 1 \quad & \text{otherwise.} \end{aligned}$$

We regard $A(m, n, \varepsilon x, \varepsilon y)$ only to involve elements with length $\leqslant n$.

We adopt the first case below that pertains.

Case 2a. For some chose ε and some x with $|x| < t$,

$$\Gamma(\Phi(\hat{A}) \oplus \Delta(\hat{A}); x) \neq \hat{A}(x),$$

where $\hat{A} = A^*(t_1, t, \varepsilon z_1, \varepsilon z_1)$.

Action. Set $A_{t_1+1} = \hat{A}$ and initialize all requirements of lower priority than $R_{e,i}$. Note that, unless this disagreement is disturbed, we have won $R_{e,i}$ (indeed R_e) forever.

Case 2b. Case 2a does not pertain but we can cause a (Φ, Δ)-*change* via z_1. That is, there exist x_1, x_2 with $|x_1|, |x_2| \leqslant \gamma(t)$ such that

$$\Phi(A', x_1) \neq \Phi(A''; x_1) \quad \text{and} \quad \Delta(A'; x_2) \neq \Delta(A'' : x_2),$$

where $A' = A^*(t_1, t, z_1, z_1)$ and $A'' = A^*(t_1, t, \neg z_1, \neg z_1)$.

Action. Apply Procedure $R(e, i, A', A'', t_1)$ below.

Procedure $E(e, i, A', A'', t_1)$. Pick $y = 0^{t_1+1}$ as a *follower* of $R_{e,i}$ targeted for Q and to be used for diagonalization against $\Psi_i(\varnothing)$. We will keep $Q \leqslant_T^p \Phi(A)$, $\Delta(A)$ as follows:

$$\begin{aligned} y \in Q \quad &\text{iff} \quad \Phi(A; x_1) = \Phi(A'; x_1) \\ &\text{iff} \quad \Delta(A; x_2) = \Delta(A'; x_2). \end{aligned} \tag{3.1}$$

We then wait until a stage $u > t_1 + 1$ such that $\Psi_{i,u}(\varnothing; y)\downarrow$ (as in the procedure $P(e, x, s)$). When stage u occurs, adopt the appropriate case below:

Case (a). $\Psi_i(\varnothing; y) = 0$.

Action. Set

$$\begin{aligned} A_u(z) &= A'(z) \quad \text{for} \quad |z| \leqslant t_1 \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

Case (b). Otherwise.

Action. Set

$$\begin{aligned} A_u(z) &= A''(z) \quad \text{for} \quad |z| \leqslant t_1 \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

Note that in either case we have $y \in Q$ iff $\Psi_i(\varnothing; y) = 0$. We initialise all lower priority requirements so that $R_{e,i}$ asserts control over $A[u]$.

Case 2c. Otherwise.

The reader should note that as neither 2a nor 2b pertain, exactly one side must change. That is, for some x with $|x| \leqslant \rho(|z|)$ it can only be that one of (3.2) or (3.3) below pertains.

$$\Phi(A'; x) \neq \Phi(A''; x) \text{ yet for all } q \text{ with } |q| \leqslant \rho(t), \Delta(A'; q) = \Delta(A''; q) \tag{3.2}$$

$$\Delta(A'; x) = \Delta(A''; x) \text{ yet for all } q \text{ with } |q| \leqslant \rho(t), \Phi(A'; q) = \Phi(A''; q). \tag{3.3}$$

If (3.2) pertains we say that $R_{e,i}$ is in state Φ (i.e., only changes in $\Phi(A)$) and if (3.3) pertains, $R_{e,i}$ is said to be in state Δ.

Comment. The reader should note that if $R_{e,i}$ enters state Φ then $\Delta(A)$ will look p-time (at least locally), since A will at the end of the construction either look like A' or A'' on strings of length $\leqslant \rho(t)$, and Δ gives the *same answer no matter which is chosen.* What we now will do, if we cannot win in the future with actions analogous to 2a or 2b, then we will ensure that the state of $R_{e,i}$ is "always Φ" and hence $\Delta(A) \in P$. The key to this is to define a new A-number z_2 with $|z_2| > t_1 + 1$

to take over the role of z_1 (and release z_1 to lower priority action, as we will see). In the basic module we will put in a step $2\frac{1}{2}$, where we define our new A-number, say $z_2 = 0^{t_1+1}$.

In the full construction, the overall machinery of the construction will eventually act to define this new number z_2; but there, unless $R_{e,i}$ is the highest priority unsatisfied requirement, $|z_2|$ will be much larger than 0^{t_1+1}; as we shall see.

3. $R_{e,i}$ will have (temporary) control over two A-numbers, z_1 and z_2, with $|z_1| < |z_2|$ and $|z_2| \leqslant t_3$ (the stage number we enter for this section of the procedure). There will be no other A-numbers that have not yet reached their final positions (i.e., in or out of A). We wait until stage $t_4 = 2^{\rho(t_3)+1}$ and then adopt the first case below to pertain.

Case 3a. For some choice $(\varepsilon_1 z_1, \varepsilon_1 z_2)$ and some x, $\Gamma(\Phi(\hat{A}) \oplus \Delta(\hat{A}); x) \neq \hat{A}(x)$, where $\hat{A} = A^*(t_4, t_3, \varepsilon_1 z_1, \varepsilon_2 z_2)$ and $|x| \leqslant t_3$.

Action. Set $A_{t_4+1} = \hat{A}$. Initialize all lower priority requirements. This meets R_e.

Case 3b. We can cause a double change. That is, there exist x_1, x_2 and choices $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4$ such that $|x_1|, |x_2| < t_3$, and

$$\Phi(A'; x_1) \neq \Phi(A''; x_1),$$
$$\Delta(A'; x_2) \neq \Delta(A''; x_2),$$

where

$$A' = A^*(t_4, t_3, \varepsilon_1 z_1, \varepsilon_2 z_2),$$
$$A'' = A^*(t_4, t_3, \varepsilon_3 z_1, \varepsilon_4 z_2).$$

Action. Win $R_{e,i}$ by applying Procedure $R(e, i, A', A'', t_4)$.

Case 3c. Otherwise. This case is the heart of the basic module. We claim that we can release z_1, and keep $R_{e,i}$ in the same state. By symmetry, we shall suppose $R_{e,i}$ was in state Φ. Inductively we will know (cf. (3.2)) that if $u(=t_1)$ was the last stage that this procedure enacted, as it finished in state Φ, then there exists x_1 with $|x_1| \leqslant \rho(|z_1|)$ such that

$$\Phi(A'; x_1) \neq \Phi(A''; x_1) \qquad \text{but that for all } q \text{ with } |q| \leqslant \rho(z_1),$$
$$\Delta(A'; q) = \Delta(A; q),$$

where

$$A' = A(u, u, \neg z_1, \neg z_1) \qquad \text{and} \qquad A'' = A^*(u, u, z_1, z_1). \tag{3.4}$$

Now we will always have chosen z_2 so that $|z_2| \geqslant 2^{\rho(|z_1|)+1}$ and, hence in particular, z_2 cannot affect the disagreement (3.4).

We claim that for any choice ε_1, if $A_1' = A(t_4, t_3, \varepsilon_1 z_1, z_2)$ and $A_1'' = A^*(t_4, t_3, \varepsilon_1 z_1, \neg z_2)$, then it must be the case that there exists x_2 with $|x_2| \leqslant \rho(|z_2|)$ such that

$$\begin{aligned} &\Phi(A_1'; x_2) \neq \Phi(A_1''; x_2) \qquad \text{and yet for all } q \text{ with } |q| \leqslant t_3, \\ &\Delta(A_1'; q) = \Delta(A_1''; q). \end{aligned} \tag{3.5}$$

Suppose (3.5) fails. Then there must exist a choice as above, where for some q with $|q| \leqslant t_3$ that we have $\Delta(A_1'; q) \neq \Delta(A_1''; q)$. Consider then

$$A_2' = A^*(t_4, t_3, \neg \varepsilon_1 z_1, z_2).$$

Now either

$$\Delta(A_2'; q) = \Delta(A_1''; q) \tag{3.6}$$

or

$$\Delta(A_2'; q) = \Delta(A_1''; q). \tag{3.7}$$

In either case we have a win. If (3.5) holds then we can apply Procedure $R(e, i, A_2', A_1'', t_4)$ and if (3.6) holds we can apply Procedure $R(e, i, A_2', A_1', t_4)$. Thus, we have (3.6). It therefore follows that for all choices for $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4$ and all q with $|q| \leqslant t_3$ that

$$\Delta(A^*(t_4, t_3 \varepsilon_1 z_1, \varepsilon_2 z_2); q) = \Delta(A^*(t_4, t_3, \varepsilon_3 z_1, \varepsilon_4 z_2); q),$$

so $R_{e,i}$ remains in state Φ.

The point of the above is this: The construction will now pick a new A-number z_3 with $|z| > t_4$, so $|z_3| > 2^{\rho(|z_2|)+1}$ and $R_{e,i}$ will release z_1 from control. The next time we enter this procedure we will use the parameters $s = t_1, z_2$, and z_3. The idea is that now a lower priority procedure might be able to control z_1 and Δ does not care whether it is in or out of A. Also the reader should note that, in the construction to follow, the next time we attend $R_{e,i}$, $A[t_1]$ will be completely decided.

At this point we would like to mention the similarity between the design of $R(e, i, s, t)$ and the central procedure of Shinoda and Slaman [20]. The key idea in both cases is that the procedure (in our case, $R(e, i, s, t)$ does not decide the value of A at a small A-number z_1 until finding a large A-number z_3 and analysing which of $\Phi(A)$ and $\Delta(A)$ reflect changes in a. The Σ_2 outcome is that changes in A are (essentially) always reflected in the *same* functional. In other outcomes we produce a disagreement between functionals that are applied to sets under construction. The fact that either the change is always on one side or a disagreement can be produced depends on the delay of the decision for z_1. A parallel analysis appears in [20]. Finally, this idea has some parallels in classical recursion theory. A similar delay is used in the central strategy of the $\mathbf{0}'''$ priority argument used in Downey [8].

The α-modules, the construction, and the coherence of the strategies. To complete the proof, it remains to give the details of how we can fit the procedures together.

To do this we have "pointers" $m(s)$ and $n(s)$. The intention is that at stage s we will be satisfying requirement $R(m(s))$ from a list of $R(-1)$, $R(0)$, ..., $R(n(s))$, where $R(0)$, $R(1)$, $R(2)$, ... is a listing of all the requirements $R_{e,i}$ and P_e, and $R(-1)$ is a new requirement that we add whose only job is to ensure that there are infinitely many A-numbers. To begin the construction at stage 0, we let $R(-1)$ act.

This means it will appoint $0^{s+1}=0$ as an A-number. In general, if $R(-1)$ acts at stage s of the construction, we assign 0^{s+1} as an A-number. We will then define $n(s+1)=n(s)+1$ (and so $n(1)=1$) and $m(s+1)=n(s+1)$.

Thereafter we will attend the currently unsatisfied requirements in ascending order of priority: $R(n(s+1))$, $R(n(s+1)-1)$, ..., $R(-1)$ in the following way: At stage t, $m(t)$ will denote the requirement we are currently satisfying. We will only be concerned with $R(m(t))$ until its procedure finished.

When $R(m(t))$'s module finishes, say at t_1, we will set $m(t_1+1)=m(t_1)-1$ unless $m(t_1)=-1$. If $m(t_1)=1$ so that $R(-1)$ acts, we will proceed as discussed earlier.

It $R(m(t))$ is P_e and P_e is not yet satisfied, we begin to examine it at stage t_1; we will let the procedure $P(l, x, t_1)$ act, where x is the longest currently defined A-number.

At the end of this procedure, we will declare P_e is satisfied and initialise all $R(j)$ for $j>m(t)$ and P_e asserts permanent control over $A[t]$. This control can only be injured (become unsatisfied) if some $R(k)$ for $k<j$ asserts permanent control over $A[t]$.

If $R(m(t))$ is $R_{e,i}$, $R_{e,i}$ is not yet satisfied, and we begin to examine it at stage t_1; we will let the procedure $R(e, i, t_2, t_1)$ act, where t_2 is determined by the construction. One difference between the α-module and the basic module is that in the full construction, if $R(e, i, t_2, t_1)$ can only act via 1 and 2, we will move on to $R(m(t)-1)$; otherwise $R(e, i, t_2, t_1)$ will act via 3. (The point is that the requirement $R(-1)$ will appoint the new A-number instead of step $2\frac{1}{2}$.)

There is, however, one other important difference between the α-module and the basic module. If at stage s, $R_{e,i}$ asserts control over, say, z_1 and z_2 and sees a win then, according to the basic module, it sees two A-configurations $A'=A(\varepsilon_1 z_1, \varepsilon_1 z_2)$ and $A''=A(\varepsilon_2 z_1, \varepsilon_2 z_2)$ which can be used to cause a double change. Furthermore, we can ask that $\varepsilon_1 z_1 = z_1$ and $\varepsilon_2 z_1 > z_1$ so that A' and A'' correspond to differing z_1-configurations. At some stage $t>s$, before $R_{e,i}$ finishes with these numbers, $R_{e,i}$ declares that some y is in Q according to $\Phi_e(A')$, $\Delta_e(A')$; or $\Phi_e(A'')$, $\Delta_e(A'')$. Note that once this axiom for the reduction is enumerated, to keep $Q \leqslant_{\mathrm{T}}^{\mathrm{P}} \Phi_e(A)$, $\Delta_e(A)$, $\Delta_e(A)$ we now must agree that the only *legal future configurations of A length t will be A′ or A″, and this agreement has priority e rather than* $\langle e, i\rangle$. *This will apply to higher priority* R_k. So now we shall let A' and A'' take the roles of z_1. That is, in the procedures we can regard the two A-configurations as taking the role of A with z_1 and A with z_1 in, respectively. So, at this stage t we will appoint another A-number $\hat{z}_2=0^{t+1}$ to be the new z_2. Thus, in effect we resurrect step $2\frac{1}{2}$. When $R_{e,i}$ releases to the R_k of higher priority it will know that it will be given an A-number z_2 and two A-configurations A' and A'' to work with. It is clear that with this modification all goes through as before.

If $R(m(t))$ acts via 2a, 2b, 3a, or 3b then it becomes satisfied and asserts permanent control over $A[t]$ and initialises all $R(j)$ for $j > m(t)$. Otherwise the control is temporary and no initialisation occurs.

The verification. To complete the proof it remains to verify that all requirements eventually obtain an environment where their satisfaction can occur. Arguing by priorities, if s_0 is a stage where $R(j)$ was initialised for the last time (and $m(s_0) > j$), if $R(j)$ asserts permanent control at any stage $t > s_0$, $R(j)$ will be met. Thus for $R(j)$ we can argue precisely as in the basic modules, and we see that all the requirements will eventually be satisfied.

REFERENCES

1. K. AMBOS-SPIES, "On the Structure of the Polynomial Time Degrees of Recursive Sets," Habilitationsschrift, Universität Dortmund, 1984.
2. K. AMBOS-SPIES, Three theorems on polynomial degrees of NP sets, *in* "Proceedings, IEEE Annu. Found. of Comput. Sci., 1985."
3. K. AMBOS-SPIES, S. HOMER, AND R. I. SOARE, Minimal pairs and complete problems, *in* "Proc. STACS 90," Lecture Notes in Computer Science, Vol. 415, pp. 24–36, Springer-Verlag, 1990.
4. A. BORODIN, R. L. CONSTABLE, AND J. HOPCRAFT, Dense and nondense families of complexity classes, *in* "IEEE Conf. Tenth Annu. Sympos. Switching an Automata Theory, 1969, pp. 7–10.
5. S. I. BRIEDBART, "The Structure of Complexity Classes and Degrees," Ph.D. thesis, University of California, Santa Barbara, 1977.
6. P. CHEW AND M. MACHTEY, A note on structure and looking back applied to the relative complexity of computable functions, *J. Comput. System Sci.* **23** (1981), 53–59.
7. S. A. COOK, The complexity of theorem proving procedures, *in* "Proceedings, Third Annual A.C.M. Sympos. Theory of Computing, 1971," pp. 151–158.
8. R. G. DOWNEY, Lattice nonembeddings and initial segments of the recursively enumberable degrees, *Ann. Pure Appl. Logic* **49** (1990) 97–119.
9. R. G. DOWNEY, W. GASARCH, S. HOMER, AND M. MOSES, On honest polynomial reductions, relativizations and P = NP, *in* "Proceedings, Fourth Annual Conference on Structural Complexity," pp. 196–207, IEEE, New York, 1989.
10. R. M. KARP, Reducibility amongst combinatorial problems, *in* "Complexity in Computer Computations" (R. Miller and J. Thatcher, Eds.), pp. 85–103, Plenum, New York, 1972.
11. R. LADNER, Polynomial time reducibility, *in* "Proceedings, Fifth Annual ACM on Theory of Computing, 1972," pp. 122–129.
12. R. LADNER, On the structure of polynomial time reducibility, *J. Assoc. Comput. Mach.* **22** (1975), 155–171.
13. L. H. LANDWEBER, R. J. LIPTON, AND E. L. ROBERTSON, On the structure of sets in NP and other complexity classes, *Theoret. Comput. Sci.* **15** (1981), 103–123.
14. K. MELHORN, The "almost all" theory of subrecursive degrees is decidable, *in* "Automata, Languages and Programming, 2nd Colloquium Saarbrücken" (J. Leckx, Ed.), pp. 317–325, Lecture Notes in Computer Science, Vol. 15, Springer-Verlag, New York/Berlin, 1974.
15. K. MELHORN, Polynomial and abstract subrecursive classes, *J. Comput. System. Sci.* **12** (1976), 147–178.
16. P. ODIFREDDI, "Classical Recursion Theory," North-Holland, New York, 1989.
17. H. ROGERS, "Theory of Recursive Functions and Effective Computability," McGraw–Hill, New York, 1967.
18. U. SCHÖNING, A uniform approach to obtain diagonal sets in complexity classes, *Theoret. Comput. Sci.* **18** (1982), 95–103.

19. U. SCHÖNING, Minimal pairs for P, *Theoret. Comput. Sci.* **31** (1984), 41–48.
20. J. SHINODA AND T. SLAMAN, On the elementary theory of PTIME degrees of recursive sets, *J. Comput. Sys. Sci.* **41** (1990), 321–366.
21. R. SHORE AND T. SLAMAN, The p-T-degrees of the recursive sets: Lattice embeddings, extensions of embeddings and the two quantifier theory, *in* "Proc. 4th Structure in Recursion Theory," IEEE, 1989.
22. R. I. SOARE, "Recursively Enumerable Sets and Degrees," Springer-Verlag, New York, 1987.
23. P. YOUNG, Some structural properties of polynomial time reducibilities and sets in NP, *in* "Proceedings, Fifteenth Annual ACM Symp. on Theory of Comput., 1983," pp. 392–401.

Printed by Catherine Press, Ltd., Tempelhof 41, B-8000 Brugge, Belgium