Downey (111-140)

NOTES ON THE $\mathbf{0'''}$ PRIORITY METHOD WITH SPECIAL ATTENTION TO DENSITY RESULTS

Rod Downey[1]
Mathematics Department, Victoria University of Wellington
PO Box 600, Wellington, New Zealand

(*e-mail* DOWNEY @ RS1. VUW. AC. NZ)

§1 INTRODUCTION

The use of trees control strategies in priority arguments has become ubiquitous in modern recursion theory. One of the crucial reasons for this seems to be that the use of trees gives us the ability to lay out the outcomes of each atomic strategy and reduces our verification to a 'coherence lemma'. That is, our strategies are so devised that Harrington's 'golden rule' is satisfied: each requirement has a version that can live with any particular sequence of outcomes of the other relevant requirements.

The power of this technique is that it often reveals the 'real reasons' that earlier results employing technical devices, such as the 'hat trick' of Soare/Lachlan (cf [16]) or Sacks density theorem (cf (2.3) of the present paper) work. In many ways the early infinite injury arguments were being described originally in a finite injury (linear) way, where in reality as $\mathbf{0''}$ arguments they would be best understood via a nonlinear tree of strategies. We remark that combinatorially, such a tree argument may be more difficult then other models(witness the use of pinball arguments for embedding nondistributive lattices into $\mathbf{R}$) but the intuition seems more readily apparent.

As yet, we do not feel the situation for $\mathbf{0''}$ and $\mathbf{0'''}$ arguments has been similarly clarified. In many ways we feel that current $\mathbf{0'''}$ arguments are really being written in a $\mathbf{0''}$ framework, and hence we decided to investigate the power of a general $\mathbf{0'''}$ framework. That is trees specifically designed for $\mathbf{0'''}$ arguments. In particular, our ideas were inspired by Shore's $\mathbf{0'''}$ argument [15] which uses an $\omega + 1$ branching tree where the true path of the construction requires a $\mathbf{0'''}$ oracle. It seemed to us that this idea ought to be general enough to encompass all $\mathbf{0'''}$ arguments and furthermore gives the potential for $\mathbf{0}^{(n)}$ arguments in a similar framework.

1. The author's research partially supported by a US/NZ binational grant.

The idea is that if we have $\omega + 1$ branches representing the outcomes of a single π_3 requirement (not spread out on the tree), then the ω outcomes $\{i : i \in \omega\}$ will be π_2 outcomes collectively giving the Σ_3 outcome and the "1" will be the π_3 outcome. Now, although one can be left of the π_3 outcome infinitely often, the π_3 outcome might be the on left most path *visited* infinitely often. In Shore's argument, this caused no problems but for general $\mathbf{0}'''$ arguments it becomes necessary to develop some machinery to allow the construction to live with this feature. Primarily one develops a 'local priority ordering' that is different from the normal priority ordering and we can allow injury from left to right and right to left.

This idea could then allow extensions to a $\mathbf{0}^{(4)}$ argument. For example, we could then use linking on the $\mathbf{0}'''$ tree, or we could use a non-uniform $\mathbf{0}'''$ argument (e.g Downey - Slaman [6]) or we could change the order type of the outcomes to be a $1 + \omega^*$sequence of $\omega + 1$ outcomes. In the last option the true path would be $\mathbf{0}^{(4)}$.

In §2 we develop the machinery in the guise of a detailed sketch of new proof of Slaman's density theorem : [14] that if $\mathbf{e} < \mathbf{f}$ then there exist $\mathbf{a} \mid \mathbf{b}$ with $\mathbf{e} < \mathbf{a}, \mathbf{b} < \mathbf{f}$ and with $\mathbf{a} \cap \mathbf{b}$ existing. It is fair to say that this is a difficult $\mathbf{0}'''$ argument and hence provides a good testing ground for the technique. Much of the discussion to (2.15) is devoted to this specific result. We remark that much of the delicacy and length of this discussion stems from the problem of keeping the construction $\leq_T F$. One of the reasons we choose Slaman's density theorem rather than an easier $\mathbf{0}'''$ argument is the fact that we feel our technique will be very good for other density questions (see §3).

We develop some general machinery in (2.15) and (2.16), and in (2.17) discuss how one could use the framework for other theorems. In particular in (2.18) we look at the Lachlan nonbounding theorem under this framework.

Finally in §3 we examine some extensions of our results. First we show that the superbranching degrees (of [4]) are dense (by observing that the strategies are compatible with §2). The other result is to use the machinery to solve a question of J.B. Remmel : We show that there exists an r.e degree $\mathbf{a} \neq \mathbf{0}, \mathbf{0}'$ such that if B is r.e and deg (B) $\geq \mathbf{a}$ then there exists an r.e splitting $B_1 \cup B_2 = B$ with deg $(B_1) = \mathbf{a}$.

Notation is standard and follows Soare [16]. All uses are monotone in stage and argument where defined, and bounded by s at stage s. We do assume the reader already familiar with tree arguments at least on the $\mathbf{0}''$ level and hope that he or she will be familiar with Soare [15, 16].

§2 A NEW PROOF OF SLAMAN'S DENSITY THEOREM

2.1 *The Components*

The goal of this section is to (eventually) give a new proof of Slaman's result that if **e < f** then there exist **a**, **b** with **a < b**, **a < f**, **a ∩ b** existing and **a | b** . In view of the fact that we shall later extend the ideas of this construction to a general approach to **0'''** arguments, we shall proceed very slowly breaking the argument into small components : streaming (and the infimum requirements below a degree), the density requirements, the coding and coherence.

Let E and F be given r.e sets with $E <_T F$. We build re sets A, B and Q to meet the requirements A, B, Q $\leq_T F$ with Q $\leq_T$ A, B and

$$P_{2e} : \Phi_e(\hat{A} \oplus E) \neq B$$
$$P_{2e+1} : \Phi_e(\hat{B} \oplus E) \neq A \text{ and}$$
$$R_e : \Phi_e(\hat{A} \oplus E) = \Phi_e(\hat{B} \oplus E) = f \text{ total} \Rightarrow f \leq_T Q \oplus E$$

where $\hat{A} = A \oplus Q$, $\hat{B} = B \oplus Q$ and $\{\Phi_e : e \in \omega\}$ is an enumeration of all procedures.

2.2 *Streaming and Fejer's Result*

The first component of our construction technique that we examine is the one we call *streaming* and should be thought of as the R_e requirements first duty : to *process* numbers into a will behaved *stream*. This is the way we meet the P_i for E = Ø, that is, prove the result from Fejer's thesis [7] that each nonzero r.e degree bounds a diamond lattice (c.f. also Lachlan [11] and Downey[3]).

Thus let E = Ø and suppose then we wish to meet the requirements above. In this case it would suffice to meet A, B, Q $\leq_T F$ and

$$\hat{P}_{2e} : \Phi_e(\hat{A}) \neq B,$$
$$\hat{P}_{2e+1} : \Phi_e(\hat{B}) \neq A \text{ and}$$
$$\hat{R}_e : \Phi_e(\hat{A}) = \Phi_e(\hat{B}) = f \text{ total} \Rightarrow f \leq_T Q.$$

By Lachlan's nonbounding theorem [10] know that Q = Ø is not always possible for all given F. Technically this means, roughly speaking, that the idea of "preserving one side of the computation between expansionary stages" fails when combined with permitting. The idea then is to meet the $\hat{R}_e$ by enumeration into Q whenever " both sides" of a computation fail between expansionary stages.

Define $\hat{l}(e, s) = \max\{x : (\forall y < x)[\Phi_{e,s}(\hat{A}_s, y) = \Phi_{e,s}(\hat{B}_s; y)]\}$ (similarly $l(e, s) \max\{x : (\forall y < x)[\Phi_{e,s}(\hat{A}_s \oplus E_s; y) = \Phi_{e,s}(\hat{B}_s \oplus E_s; y)]\}$) $m\hat{l}(e, s) = \max\{0, \hat{l}(e, t) : t < s\}$ (similarly $ml(e, s)$), $m\hat{u}(e, s) = \max\{u(\Phi_{e,s}(\hat{A}_s; x)), u(\Phi_{e,s}(\hat{B}_s; x))\}$.

The basic idea to meet the R_e is to allow the R_e to create a well behaved *stream* of numbers as follows. R_e will be given a set (ω for the basic module, the numbers processed by higher priority R_j (in the α - module) that it will *process* as follows. Initially it is given x_0 (say 0). At the first stage s_0 where $l(e, s_0) > x_0$, R_e will process x_0 and pick a new number x_1 with $x_1 > m\hat{u}(e, x_0, s_0)$ (eg $x_1 = s$), and from the point of view of cooperation with other requirements, cancel (or restrain) all numbers z with $x_0 < z < x_1$ and so stop them from entering A.

Remark Variations are possible here. For example, we can use a "dump" construction and ask that these numbers enter A iff x_0 enters A at the same stage.

In general we continue this process, and, assuming nothing has yet been enumerated, we will have at any stage a stream of numbers $x_0 < x_1 < x_2 < \_\_ < x_n$ such that each $x_i > m\hat{u}(e, x_j, s)$ for all $j < i$ (recall we assume uses monotone where defined).

What is the point of this procedure? The first thing to notice is that x_{i+1} is "good" for x_j for $j \leq i$ in the sense that if x_{i+1} enters $\hat{A}$, $\hat{B}$ or both, *it will not affect the computations for* x_j *for* $j \leq i$. The idea then is for those versions of $\hat{P}_j$ (guessing that $\hat{R}_e$'s effect is infinitary) of lower priority than $\hat{R}_e$ will use only numbers from the $\hat{R}_e$ - stream for followers, and themselves will also process this stream in essentially the same way.

Thus to meet $\hat{P}_{2j}$ (for example) $j > e$, $\hat{P}_{2j}$ will take $x_0 = x(2j, 0, s)$ and in a similar way, wait till $\hat{L}(2j, s) > x(2j, 0, s)$ where

$$\hat{L}(2j, s) = \max\{x : \forall y \leq x(\Phi_{j,s}(\hat{A}_s; y) = B_s(y))\}$$
$$\hat{L}(2j+1, s) = \max\{x : \forall y \leq x(\Phi_{j,s}(\hat{B}_s; y) = A_s(y))\}.$$

At this stage (when we see $\hat{L}(2j, s) > x(2j, 0, s)$) $\hat{P}_{2j}$ will request the next member of $\hat{R}_e$'s stream and cancel/restrain all current members of $\hat{R}_e$'s stream $> x_0$. At the stage that we see a new member of $\hat{R}_e$'s stream appear, say x_n, we will assign $x(2j, 1, s) = x_n$ etc. Thus the first action of $\hat{P}_{2j}$ is to *refine* $\hat{R}_e$'s stream to look like

$$x(2j, 0, s) = x_0, \underbrace{x_1, x_2 \_\_}_{\text{don't really exist}}, x_n = x(2j, 1, s), \underbrace{x_{n+1} \_\_}_{\text{nor do these.}} x_m = x(2j, 2, s), \_$$

Now $\hat{P}_{2j}$'s final act is to wait till we see F permit i at some stage s where x(j, i + 1, s) is defined. When this occurs, we wish to enumerate x(2j, i, s) into B_i so winning $\hat{P}_{2j}$ (a Friedberg - Muchnik procedure).

The problems stem from the fact that other $\hat{P}_k$ with k > e *also* must select their x(k, n, t) from R_e's stream. Arguing by priorities, at any stage the R_e - stream will have been refined to look like

$$x(2e, 0, s), \_, x(2e, n_{2e}, s), x(2e + 1, 0, s), \_, x(2e + 1, n_{2e+1}, s), \_\_\_,$$
$$x(m, 0, s), \_\_, x(m, n_m, s) \_\_$$

where the x (k, d, s) are devoted to R_k awaiting F to permit d. The bad scenario is that at some stages for some k we see F permit k and so we might enumerate (say) x(2r + 1, k, s) into A_{s+1} to win R_{2r+1} for some r > j. Since we did not win (e.g.) R_{2j} it must have been that x(2j, k, s) was not defined. But now, at some later stage we see F permit i and we wish to enumerate x(2j, i, t) =(2j, i, s) into B_{t+1} to win R_{2j}. The problem is that our enumeration of x(2r + 1, k, s) into the A-side earlier might have destroyed some A-computation and if we now enumerate x(2j, i, t) we might *also destroy the B-side of the same computation.*

The *solution* is to *also* enumerate x(2r + 1, k, s) into B_{t+1} and Q_{t+1} at the same stage as we enumerated x(2j, i, t) = x(2j, i, s) into the A-side. In this way Q can comprehend the fact that both sides may have changed. Notice that at least one side remains valid for x_q < x(2r + 1, k, s). For our purposes, it is worthwhile to think of this as follows. If the R_e - stream looks like x(e, 0, s), x(e, 1,s), ___ at any stage, then we regard x(e,n + 1, s) as the Q-use for x(e, n, s). Hence if x(e, n + 1, s) is enumerated into Q (for example, x(e, n + 1, s) = x(2r + 1, k, s) as above) then at the next e-expansionary stage t we could redefine x(e, n + 1, t).

It is not hard to see that - in this construction - we can ensure that if $l(e, s) \to \infty$ then $\lim_s$ x(e, n, s) = x(e, n) exists. Then suppose $\Phi_e(\hat{A}) = \Phi_e(\hat{B}) = f$ with f total. Let z be given. Note that x(e, z, s) > x for all z, s. Now find the least e-expansionary stage where x(e, z + 1, s) is defined and x(e, z + 1, s) $\notin$ Q. Then $\Phi_{e,s}(\hat{A}_s ; z) = \Phi_e(\hat{A}; z)$.

The remaining details are to implement the above for all $\hat{R}_e$ and $\hat{P}_e$ via a tree of strategies. Note that all of the actions are compatible as each $\hat{R}_e$ and $\hat{P}_e$ essentially wishes to do the same thing : refine a set of numbers they are given into a well behaved stream. The only conflict ocurs due to the fact that we wish to make sure that $(\forall i)(\lim x(e, i, s)$ exists), but in the present argument this causes no problems since such injury is only caused *by us via the finitary requirements* $\hat{P}_i$. Similar arguments that involve processing numbers can be combined with the strategy above. For example, it is not difficult to extend the ideas above to embed the countable atomless boolean algebra below any nonzero r.e degree ([3]), or to construct a *superbranching* degree (i.e. $a \neq \mathbf{0}'$ such that $\forall\, \mathbf{b} > \mathbf{a}\ \exists\ \mathbf{c}, \mathbf{d}\ (\mathbf{a} < \mathbf{c}, \mathbf{d} < \mathbf{b}\ \&\ \mathbf{c} \cap \mathbf{d} = \mathbf{a})$). Downey - Mourad [4]) or finally to construct a contiguous nonbranching degree (Downey [2]). (This last paper and [5] were the places where the terminology and ideas were developed).

We remark that the above streaming is a little too simple minded for Slaman's density theorem due to the interaction of E-coding and P_j action as we shall see, but the ideas really underpin the construction.

2.3 *The Density Requirements*

We return to the Sacks requirements

$$P_{2e} : \Phi_e(A \oplus E) \neq B,\ P_{2e+1} : \Phi_e(B \oplus E) \neq A.$$

When Sacks density theorem was first proved, it was apparently accomplished by a series of clever tricks one of which is the famous Sacks coding strategy. Today, using tree of strategies it is possible to expose the underlying intuition behind this strategy, and to see that it is really no more than the original Friedberg - Muchnik method and the *delayed permitting method.*

In this section we let $Q = \emptyset$ so that $\hat{A} = A$ and $\hat{B} = B$.

Suppose that also $E \equiv_T \emptyset$ and so we needed to only build $A, B \leq_T F$ so that $\Phi_e(A) \neq B$ (and $\Phi_e(B) \neq A$). In this case we would the a familiar process of the last section : define a stream of followers $x(2e, i + 1, s)$ for $i \leq n = n(s)$ so that $x(2e, i + 1, s)$ exceeded the use of $x(2e, i, s)$, we would wait till $i \in F_{\text{at } s}$. If this occurs, we'd enumerate $x(2e, i, s)$ into B winning R_{2e}.

If $E \not\equiv_T \emptyset$ the problem is that E can later code numbers below the use of $x(2e, i, s)$ to upset this win. For a single requirement P_{2e} our solution is to define our stream to essentially obey the following rules (2.4) - (2.6).

2.4 ***Cancellation*** If $x = x(2e, i, s)$ is currently *active* (that is, waiting for an F-permission and $x(2e, i+1, s)$ is defined) and we see that $\Phi_{e,s}(A_s \oplus E_s ; x)$ is E-incorrect cancel $x(2e, j, s)$ for $j > i$. If $x(2e, i, s) \in B_s$ also cancel $x(2e, i, s)$ and declare $x(2e, i, s)$ asinactive.

2.5 ***Activation (Appointment)*** If $x(2e, i, s)$ is currently defined but not active (and hence $x(2e, i+1, s)$ is not defined) and $L(2e, s) > x(2e, i, s)$ set $x(2e, i+1, s) > s$, $r(2e, s+1) = s$ and declare $x(2e, i, s)$ as active.

2.6 ***Permission*** If $x(2e, i, s)$ is active and $i \in F_{\text{at } s}$ enumerate $x(2e, i, s)$ into B_{s+1} cancel $x(2e, j, s)$ for $j > i$ but regard $x(2e, k, s)$ for $k \leq i$ as active.

The reader should note that, because of the last clause of (2.6), P_{2e} can still receive attention (via some $x(2e, k, s)$ for $k < i$) whilst it appears satisfied, as this attack (via k) is more likely to succeed. The crucial point is that whilst P_{2e} appears temporarily satisfied (via $x(2e, i, s)$) it *cannot get any new followers appointed to it.*

The rules above suffice for a single P_{2e}.

2.7 ***Lemma*** *Suppose* $(\forall x)[\Phi_e (A \oplus E ; x)\downarrow]$. *Then* $(\exists y)\ [\Phi_e (A \oplus E ; y) \neq B(y)]$ *and* P_{2e} *acts only finitely often.*

Proof Suppose not. We show $F \leq_T E$. It suffices to show that

(a) $(\forall s)(\lim_s x(2e, i, s) = x(2e, i)$ exists and $\notin B)$

(b) $(\forall i)(\exists s)\big(x(2e, i, s) = x(2e, i)$ and $l(2e, s) > x(2e, i)$ and $u(\Phi_{e,s}(A_s \oplus E_s ; x(2e, i)) = u(\Phi_e(A \oplus E ; x(2e, i))\big)$

(c) $(\forall s)\big(x(2e, i+1, s) > u(\Phi_{e,s}(A_s \oplus E_s ; x(2e, i, s))\big)$.

(d) E can recognise when (a) and (b) occur.

Once we have (a) - (d) we can E-recursively compute F as follows. Let $z \in \omega$. E-recursively find a stage s where $x(2e, z+1)$ is defined. Then $x(2e, z)$ is active and for all $j \leq z$ the Φ_e - computation on $x(2e, j)$ are E-correct. By restraints, the computations are final. Hence $z \in F$ iff $z \in F_{\text{at } s}$ since otherwise z's entry into F would cause us to win P_{2e} at or below $x(2e, z)$.

To verify (a) - (d), suppose we have already computed $x(2e, 0)$, __, $x(2e, k)$ and a stage s_0 where $\forall s \geq s_0\ (x(2e,j) = x(2e, j, s))$ for $j \leq k$. By hypothesis (a) - (d) hold for

$j < k$: Note $x(2e, k) \notin B_{so}$ otherwise the $\Phi_{e, so}(A_{so} \oplus E_{so} ; x(2e, k))$ computations are E-incorrect (so that $x(2e, k)$ not final a contradiction). Now E-recursively find a stage s_1 where $L(2e, s_1) > x(2e, k, s_1)$ via E-correct computation. Then $x(2e, k + 1, s_1 + 1) = x(2e, k + 1)$.

Thus the method above solves the problem for a single P_{2e}. A problem is caused by the interaction of (2.5) and (2.6) for the coherence of several P_j. Our concern is as follows.

Suppose that for some least k we have $L(2e, s) \to \infty$ but $\Phi_e(A \oplus E ; x(2e,k)) \uparrow$ so that the use $\to \infty$. Now the $x(2e, j)$ for $j < k$ have finite effect, but infinitely often the $x(2e, j, s)$ - list is chopped back to $x(2e, 0), \_, x(2e, k)$ (a π_2 - event). Each time this is chopped back, at the next e-expansionary stage, we reset $r(2e, t)$ to $u\left(\Phi_{e, t}(A \oplus E ; x(2e,k))\right)$. We thus meet P_{2e} by divergence.

Consider some P_j for $j > 2e$ desiring to put some numer into A. Such P_j can only put unrestrained numbers into A, and if P_{2e} is met by divergence then potentially its restraint is infinite. The usual way dilemmas like the above are overcome is to allow the restraint to "drop back" at E-nondeficiency stages a'la Soare [16]. In our case we need F-permitting too (and this is the heart of Sacks trick). The bad scenario is as follows. We have some $\hat{x} = x(j, p, s)$ targeted for B. We see F permit $\hat{x}$ at stage s but at this stage $r(2e, s) > \hat{x}$. It may be the case that at some stage $t > s$ it is found that $r(2e, t)$ is E-incorrect and the restraint drops. But we have no longer E-permission to allow us to put $\hat{x}$ in.

The key modification is to replace (2.6) by *delayed permission* The crucial point is that E knows if $r(2e, s)$ is E-correct or not (remember $r(2e, s)$ only drops due to E-incorrect computations (essentially).). Further as $E \leq_T F$, whatever E knows, F knows too. This when we see $\hat{x}$ F-permitted we *declare it so*. Now, whilst $\hat{x}$ is still active and F-permitted, should we discover $r(2e, s)$ to be E-incorrect and so drop back we then allow $\hat{x}$ to enter A. The whole point is that F can still decide the fate of $\hat{x}$ and so the construction remains $\leq_T F$.

The implementation of the above on a tree of strategies is as follows. A node σ devoted to P_{2e} has $\omega + 1$ many outcomes labelled from left to right
(0,u),(0,d),(1,u),(1,d), _ _ _, ω where

(i,u) denotes unbounded use as $x(2e, i)$

(i,d) denotes disagreement preserved at $x(2e, i)$

w denotes wait (i.e $L(2e, s) \not\to \infty$).

A version of P_j guessing (0,u) will be guessing that all the $\Phi_{e,s}(A_s \oplus E_s ;x(2e, 0))$ computations are E-incorrect (whose $\lim_s x(2e, 0, s) = x(2e, 0)$ exists). Thus a follower of this version of P_j in some sense "doesn't believe" an F-permission until it sees the Φ_e restraint drop to zero. Note that although F *cannot decide* which is the correct *outcome,* for any follower x, F can decide if the conditions that will allow x to be enumerated will ocur.

For example if a follow x of P_j had guess $\sigma = (0,u)\wedge(1,d)\wedge(5,u)$, then we would permit x to enter the relevant set provided that we saw σ appear correct (again), which would happen by the next E-nondeficiency stage, if it will happen at all.

The ideas embodied in Sacks delayed permitting are really at the heart of our actions in the full construction.

2.8 *Coding and Infimum*

In this section we shall see what happens when we combine (2.2),(2.3) and the coding of E into the infimum. It is here the argument gets complicated and becomes at **0'''** one. We will essentially meet the P_j as we did in (2.3) but must re-examine the infimum requirements.

$$R_e : \Phi_e (\hat{A} \oplus E) = \Phi_e(\hat{B} \oplus E) = f \text{ total} \Rightarrow f \leq_T Q \oplus E.$$

In (2.2) with $E \equiv_T \emptyset$ we showed how to meet Friedberg type requirements in the presence of such R_e whilst keeping the sets $\leq_T F$. The idea was to use Φ_e to process numbers into a well behaved stream, x_0, x_1 ___ so that $x_{i+1,s} > mu(e, x_i, s)$ when appointed, and hence $x_{i+1,s} >$ one of the uses whilst $x_{i+1,s}$ is not enumerated. The P_j predicated on infinitary R_e beavior then choose numbers from the R_e - stream and this allows us to enumerate x_{i+1} into both sides should come x_j for $j \leq i$ be enumerated into one side.

In fact it is easy to combine this strategy with the $\hat{P}_j$ and hence show that for all $\mathbf{e} < \mathbf{f}$ there exist **a**, **b** such that $\mathbf{a} \cup \mathbf{e} \mid \mathbf{b} \cup \mathbf{e}$, $\mathbf{a} \cap \mathbf{b}$ exists and $\mathbf{a}, \mathbf{b} < \mathbf{f}$. All of our problems stem from E-coding in the infimum requirements.

Note that E-coding in the R_e might cause infinitary behavior for two reasons. First the use on some side for some y might be unbounded (as with a P_j). The second reason is that the use might be bounded for all y but $l(e,s) \rightarrow \infty$ (the π_3 reason).

Our hope is to develop some reasonably general machinery to deal with such π_3 situations.

2.9 *The Basic (R_e -) Module*

The main idea is to, along the lines of Shore [15], represent R_e by a single outcome on the tree. Specifically σ devoted to R_e has $\omega + 2$ (primary) outcomes labelled (o, u),(1, u), ___, b, w where.

(i, u)	denotes *unbounded* use at x_i (on either A - or B- side)
b	represents the π_3 outcome *bounded* use for all x_i and $l(e, s) \to \infty$
w	represents the *waiting* "non-recovery of computation" outcome.

Note that the collection of π_2 outcomes $\{(i, u) : i \in \omega\}$ forms the "Σ_3 outcome". The idea is that the leftmost math *visited* infinitely often will be on the tree path (TP). The problems will stem from the combinatorics. It is possible that σ^b $\subset$ TP and yet in the construction we are left of σ^b infinitely often.

For the *basic module* (for R_e) the idea is to play outcome (i, u) whenever s is a σ - stage and one of the computations $\Phi_{e,s}(\hat{A}_s \oplus E_s ; x_{i,s})$ or $\Phi_{e,s}(\hat{B}_s \oplus E_s ; x_{i,s})$ prove to be E-incorrect since the last σ-stage. It will be the case that we will have enumerated $x_{i+1,s}$ into Q so that Q can comprehend this fact. Note that if has a limit x_i then if σ^(i, u) $\subset$ TP, it must be that $\Phi_e(A \oplus E, x_i) \uparrow$.

We must make sure that σ^(i, u) produces a stream of good numbers or those $\gamma \supset \sigma$^(i, u) should σ^(i, u) be correct. Thus σ produces many types of streams. It will be given a stream $\{x(\sigma^+, i, s) : s, i \in \omega\}$ where σ^+ ^a = σ and will refine this stream.

The final σ-stream $\{x(\sigma, i, s) : i \in \omega\}$ will depend on the outcomes of σ. In particular, we focus on the outcomes σ^(i, u) and σ^b. To indicate that those are two possible types of stream produced for these outcomes, we will write them as y(σ^a, j, s) where a = (i, u) or b. Each outcome (i, u) has associated with it a number p(i, s) to test it. The idea is that y(σ^b, 0, s) = x(σ, 0, s) =y(σ^(k, u), 0, s) all s and p(0, s) = 0 but that y(σ^b, i, s) = x(σ, p(i, s), s) for some p(i, s) $\geq$ i (with p(i + 1, s) > p(i, s)). Initially we will define p(i, s) = i. When we get to first define y(σ^b, i, s) we will have defined y(σ^b, i, s) =

$x(\sigma, p(i, s), s)$. Suppose this is its limit value and write $x_i = y(\sigma\hat{}b, i, s)$. Now at the next e-expanssionary stage s we will define $x_{i+1,s} = y(\sigma\hat{}b, i+1, s) = x(\sigma, p(j+1, s), s)$ (=s (for the basic module)) which will equal $y(\sigma, p(j, s)+1, s)$ if this is the first stage so that $x_{i+1,s} > mu(e, x_i, s)$. Now this would be a $\sigma\hat{}b$ - stage and playing $\sigma\hat{}b$ we'd be free to assign these numbers to P_j for P_j guessing $\sigma\hat{}b \subset TP$.

If, at a later stage t we see that the current Φ computations on x_i were not both E-correct (this is slightly modified when we consider the interaction with the various P_j), we should have enumerated $x_{i+1,t} = x(\sigma, p(k+1, s), s) = x(\sigma, p(k+1, t), t)$ into Q cancelling any P_j restraint working with $x_{i+1,t}$. At the next e-expansionary stage t_1, we would then play a $\sigma\hat{}(i, u)$ stage.

One notable point here is that such $x_{i+1,t}$ need to be enumerated as soon as they appear E-correct. This is because the construction must remain $\leq_T F$ and hence we cannot wait until t_1. We must however wait until t_1 to *appoint* the next $x_{i+1,t}$ as we now see.

At this stage we would set aside a collection of numbers $C = \{x(\sigma, p(k)+i, t_1), \_\_\_ x(\sigma, p(k)+t, t_1)\}$ (say) and then assign $p(k+1, t_1) = p(k)+t+1$ and $x(\sigma\hat{}b, i+1, t_1) = x(\sigma, p(k)+t+1, t_1)$.

The "back-up" stream is $\{y(\sigma\hat{}(i, u), j, s : j \in \omega\}$. This would have the collection C added to it (in order). Such numbers can only become followers of nodes $\gamma \supset \sigma\hat{}(i, u)$ and only be *assigned* during $\sigma\hat{}(i, u)$ stages. Since we need the sets must be $\leq_T F$, it is necessary that they enter during other than $\sigma\hat{}(i, u)$ stages. These elements will have a finite collection of restraints they must respect. Obviously won't respect any restraint based on $x_{j,s}$ for $j \geq i+1$, but as we shall see, they may respect restraints based on $x_{k,s}$ for $k \leq i$. These restraints nevertheless behave in a friendly fashion (as in (2.3)) and drop down at E-nondeficiency stages. This allows us to decide if such a follower will enter.

Note that, in the limit, if $\sigma\hat{}(i, u) \subset TP$ then all of the σ - stream is of the form $y(\sigma\hat{}(i, u), j, s)$ and all the $y(\sigma\hat{}b, k, s)$ die (for $k > i$).

On the other hand if $\sigma\hat{}b$ is correct then eventually we will stop building $y(\sigma\hat{}(i, u), k, s)$ as the uses of $y(\sigma\hat{}b, i)$ come to a limit.

2.10 *Coherence of One P_j of Lower Priority with an R_e*

There are two types of P_j we must consider : those associated with $\rho > \sigma$^(i, u) and those associated with $\tau > \sigma$^b. When to play the ρ versions is clear enough (at present) namely when y(σ^b, i + 1, s) appears incorrect, and such a ρ will act during such "gaps". (A delicate part of the construction is to keep the F-recursive, but this is like the delayed permitting part of (2.3), as we see later).

Where we do have some real potential coherence problems is with a P_b associated with some $\tau \supset \sigma$^b.

Suppose that such a P_b targets an element x(τ, n, s) for B and that $x_{k,s}$ = y(σ^b, k, s) = x(τ, n, s).

Now we wish to implement the strategy of (2.3). Recall this was given in the rules (2.4) - (2.6) and basically consisted of a follower x being initially inactive, then active and then eventually perhaps enumerated after an E-correct permission. Note that the activation of x will probably occur after $x_{k+1,s}$ has been appointed.

Here we are also committed to something like the processing of (2.2). Thusx(τ, n + 1, s) will be appointed after $x_{k,s}$ activated, but x(τ, n + 1, s) is probably larger than $x_{k+1,s}$ and indeed perhaps $x_{k+1,s} < u(\Phi_{j,s}(A_s\,;\,x_{k,s}))$ where j = e(τ).

The reader should note that in the discussion to follow, the difference between on situation here and (2.2) is that in (2.2) the only change to such a set-up is due to P_j activity, and in the basic module for R_e, the only change in $x_{k+1,s}$ is due to E-coding making a Φ_e - computation E-incorrect. Here we are concerned with the *combination* of such events. If E-coding destroys such an $x_{k,s}$ = x(τ, n, s) - set up before we add $x_{k,s}$ to B we have no problems.

The bad situation is the following *(which does not occur in* (2.2)*)*. Suppose at some stage s_1 we enumerate $x_{k,s}$ into B_{s_1}. This will cause the e = e(σ) computation concerning $x_{k,s}$ to be destroyed. Consider the situation at the next e-expansionary σ-stage t_1. This is possibly as given in the diagram 1 below.

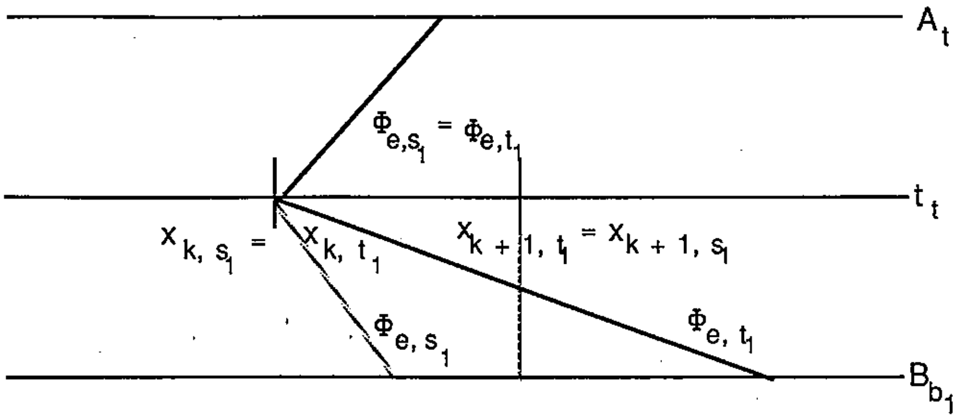


Diagram 1

In particular $x_{k+1,t_1} = x_{k+1,s_1} < u = u(\Phi_{e,s_1}(B_{t_1}; x_{k+t_1}))$. This causes the following potential problem : Remember the verification of (2.7). This consists of arguing that should we not win (e.g.) P_b then $F \leq_T E$. We do so by arguing that once we see an E-correct set up on X_{k,s_1} F cannot permit n lest we win on x_{k,s_1}. Now the reader should note that in the argument of (2.3) this followed as the only way that x_{k+1,t_1} entered into A or B and so $\hat{A}$ would be because of some $\hat{P}$'s activity. Here also E-coding for R_e via σ can cause problems via such enumeration.

Certainly in the definition of "E-correct set up for $x_{k,s}$" we can ask that for all $x_{m,s}$ if $x_{m,s} < u_{1,s_1}(\Phi_{j,s}(A_{s,j}\, x_{k,s}))$ then $x_{m,s}$ has currently E-correct Φ_e-computations too. This is consistent with $\tau > \sigma$^b after all.

Nevertheless, in the diagram above, although Φ_{e,s_1} was E-correct for $x_{k,s_1} = x_{k,t_1}$, by enumerating x_{k,s_1} into B we can cause Φ_{e,t_1} to have a much bigger use at t_1. Suppose that E now decides to cause a change in B below u_{1,s_1} *(but not below* mu(e, x_{k,s_1}, s_1)). If we now implement the basic module for R_e as stated, we must enumerate x_{k+1,s_1} into Q and so code it into $\hat{A}$ causing us to perhaps destroy the $\Phi_{j,s_1}(A_{s_1}; x_{k,s_1})$ computations. But now lemma (2.7) fails. Although $\hat{A}$ was E-correct for all potentially injurious numbers (and so $\hat{B}$ was E-correct on mu(e, x_{k,s_1}, s_1)) B was not E-correct on u_{1,s_1} allowing F to permit n late.

Our solution here is to note that, after all, the common value f can only change if *both sides* of the Φ_e - computation change. This, although we should clearly play on outcome $\sigma\hat{}(k, u)$ and reset $p(k+1, t_1)$ (to - in particular - exceed u_{1,s_1}) we should not enumerate x_{k+1,t_1} unless *both* sides change and hence, in particular if $\Phi_{e,s_1}(A_{s_1}; x_{k,s_1})$ is E-correct. Thus with this modified form of the basic module, if $\Phi_{e,s_1}(A_{s_1}; x_{k,s_1})$ proves to be E-incorrect then it might cause a couple of numbers to enter (e.g. x_{k+1,s_1} and $x_{p(k+1,t_1),t_1}$).

The idea is then that when we verify P_b at τ we go to a stage s where $x_{k,s} = x(\tau, n, s)$ is E-correct for both e and j computations and argue as in (2.7) that $n \in F$ iff $n \in F_{\text{at } s}$.

2.11 *Coherence of R_e with Two P_j (of lower priority)*

The "α - module" for R_e of (2.10) was okay for one P_j, but the problems become more subtle for two P_j. Let P_a and P_b be two such requirements and suppose that P_a has guess γ and P_b as before has guess τ, P_a targets for A, (P_b for B) and that for simplicity $\sigma\hat{}b < \gamma < \tau$.

Suppose that $e(\gamma) = m$ and that we are concerned with a follower of P_a of the form $x_{r,s} = x(\gamma, d, s)$ and note that (by priorities) $r < k$.

Now if $x_{r,s}$ was active before $x_{k,s}$ was enumerated, then the priority set up will ensure that $x_{k,s} > x(\gamma, d+1, s)$ and so, should we enumerate $x_{r,s}$ into A we can enumerate (safely) all $x_{t,s}$ for $x_{t,s} \geq x(\gamma, d+1, s)$ and so reduce this case to the basic module.

Note that we are really here considering the α - correct version of (2.2) : what can we do so that Q can comprehend the fact that two sides have changed? The method of (2.10) works in (2.2) and works for the case above.

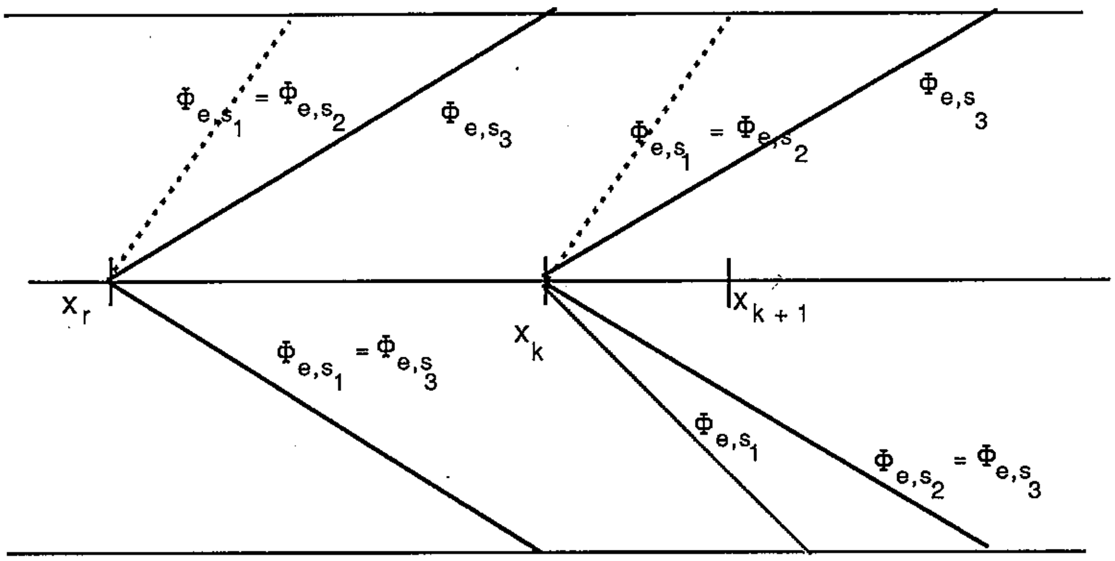
However a case that is seriously different (due to E-coding) is the following sequence:

(2.12) $x_{k,s}$ enumerated at s_1
(2.13) $x_{r,s}$ is activated at the next γ - stage s_2
(2.14) $x_{r,s}$ is enumerated at $s_3 > s_2$.

Now in (2.2) although we know that since (2.13) occurred after (2.12), it did occur *at a σ-expansionary stage.*

Hence, although $x_{k+1,s}$ might be this below $u_2 = u(\Phi_{m,s_2}(B_{s_2}; x_{r,s}))$ this doesn't matter since now the B-side will hold the computation (in (2.2) arguing by priorities) that there are no followers z left alive with $x_{r,s} < z < u_2$).

The situation is given in diagram 2 below.



$\Phi_{e,s_1} = \Phi_{e,s_2}$
Φ_{e,s_3}
$\Phi_{e,s_1} = \Phi_{e,s_2}$
Φ_{e,s_3}
x_r
x_k
x_{k+1}
$\Phi_{e,s_1} = \Phi_{e,s_3}$
Φ_{e,s_1}
$\Phi_{e,s_2} = \Phi_{e,s_3}$

Diagram 2

Note that it is now possible for both sides of the x_k - computation to have changed. E-coding causes the following problem to the (2.7) argument.

We are forced to enumerate x_{k+1} into Q should E permit both sides after s_3. Indeed, note that in the situation above, it is also possible for x_{k+2,s_2} to affect x_{k+1} as perhaps $x_{k+2} < q_1 = u(\Phi_{e,s_3}(A_{s_3}; x_{k+1}))$, $q_2 = u(\Phi_{e,s_3}(B_{s_3}; x_{k+1}))$ for example E might permit q_1 and q_2 causing us to enumerate x_{k+2} which might be below both $u(\Phi_{e,s_2}(A_{s_2};x_k))$ and $u\Phi_{e,s_2}(A_{s_2}; x_k))$ so that we are forced to enumerate x_{k+1} by a cascade effect (Diagram 3)

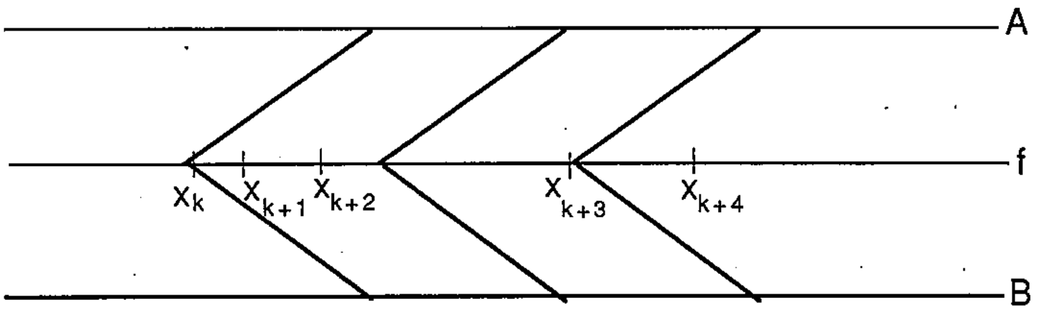


Diagram 3 : A cascade effect with x_{k+4} ***causing*** x_{k+1} ***to be enumerated***

Because of the fact that $x_{i+1,s} > mu\ (e, x_i\ s,)$ we know that the only way this situation can occur is for two numbers like x_k and x_r to enter B and A, in the sequence described above.

Again it is possible for $x_{k+1} < u_2$. We must still be above to rescue (2.7) as with (2.10). Evidently now we must ask that all computations 'potentially injury' u_2 must be E-correct (at activation). This in particular, in the verification, γ would demand that if $\Phi_{e,s_2}(B_{s_2}; x_{k+1})\downarrow$ then $u(\Phi_{e,s_2}(B_{s_2}; x_{k+1}))$ is E-correct and so on up the cascade. The whole point is that *at the time of activitation* of $x_{r,s}$ (in (2.13)) γ can see all these computations. The only difficulty then is to convince oneself that for all i, there is a γ-state where all computations potentially injury $u_2(x(\gamma, i, s))$ are E-correct should σ^b $\subset$TP and $\Phi_m(\hat{B} \oplus E) = \hat{A}$. The reason that such a stage must occur is that as soon as $x(\gamma, i, s)$ is activated, it asserts control on all existing $x_{g,s}$ for $x_{g,s} > x(\gamma, i, s)$ *and stops them being followers.*

Thus the only way they will enter Q is if E forces them to enter. Now if $i \notin F - F_s$ then $x(\gamma, i, s) = x(\gamma, i)$ will not enter A and if $\Phi_m(\hat{B} \oplus E) = \hat{A}$ then the m - use of $x(\gamma, i)$ is bounded. Computations that are not E-correct cause enumeration and hence as $\Phi_{m,s}(\hat{B}_s \oplus E_s; x(\gamma, i))\uparrow$ only finitely often, such a stage must occur. Thus (2.7) can be rescued.

Another subtle point is that the construction must be kept $\leq_T F$. The situation above has the potential to cause mischief since now perhaps large E-changes can cause small enumeration, i.e. a relatively large change in E can cause a relatively small number like x_{k+1} to enter Q (perhaps $u_3 = u(\Phi_{e,s_2}(B_{s_2}; x_k))$ and $u_4 = u(\Phi_{e,s_3}(B_{s_3}; x_k))$ are very big.

For the basic R_e-module F had no problems as $x_{k+1,s} \in Q$ iff the $x_{k+1,s}$ is appointed to trace $x_{k,s}$ and either computation is E-incorrect. The way traces will be appointed will ensure this is $\leq_T F$.

In our case the procedure is as follows. F knows that any number to enter is either an active follower or a trace. In the situation above the procedure goes roughly like this. To decide if $x_{k+1,s}$ enters Q first ask if the e-computations of $x_{k+1,s}$ are E-correct. If not then $x_{k+1,s} \in Q$. If so then note that they only way $x_{g,s}$ for $g \leq k+1$, s can enter will be because they are followers.They will be of the form $x(\rho, i, s)$ for some $\rho > \sigma$^b. Find the stage s_4 where F stopes permitting all such i. If $x_{k+1,s} \notin Q_{s_4}$ then no $x(\rho, i, s)$ which was active *before* $x_{k+1,s}$ *was appointed* can have entered (as for such $x(\rho, i, s)$ we would enumerate $x_{k+1,s}$ as we mentioned before). Thus the only $x(\rho, i, s)$ to have entered after s would be ones which became active *at* σ^b-*stages after s.*

Let $s_5 \leq s_4$ be the stage where the last such $x(\rho, i, s)$ entered. By the argument above (for the α-module) since we won't enumerate any existing x_{g,s_5} unless both sides change (and be E-incorrect) we see that $x_{k+1,s} \in Q$ iff $x_{k+1,s} \in Q_{s_6}$ where $s_6 = \mu s(E[s_5] = E_s[s_5])$. In this way, A, B and $Q \leq_T F$.

2.15 The Priority Ordering and the Problems with $\omega + 2$ Branches : General Machinery

In a normal **0''** priority construction one defines a priority ordering $\leq_p$ via lexicographic ordering on the tree of strategies (eg Soare [15, 16]). Then if σ is played during stage s we would initialise all τ to right of σ. The argument is that if σ is the leftmost math visited infinitely often than we are *left of* σ *only finitely often*. This is no longer true in our construction. Here we have $\omega + 2$ branches, and, should b be the correct outcome we might nevertheless actually be left of b infinitely often. Were we to use the standard **0''** intialization strategy (i.e. initialize nodes right of τ when we vist τ), we would intialise all $\gamma \supset \sigma$^b cofinally in the construction.

Thus we will be guided by the principle that we cannot intialise all of $\gamma \supset \sigma$^b each time we are left of σ^b. On the other hand, $\beta > \sigma$^(k, u) cannot respect all of the$\gamma \supset \sigma$^b when we play σ^(k, u). There are two reasons for this. First - many such γ would be using $x_{\hat{k},s}$ for $\hat{k} > k$ which would 'appear wrong' when we visit σ^(k, u). Second, from more general grounds, σ^(k, u) ought to only respect a finite number of nodes extending σ^b, so as to be above to be met if σ^(k, u) $\subset$ TP.

Our idea is to define a (local) priority ordering $<^*$ so that $\tau <^* \gamma$ implies γ must respect τ's restraints. We delay the exact definition of τ until later, but we ask that $<^*$ is a well ordering.

To motivate the following, consider a simple situation where $\sigma = \lambda$ and so we ask what sort of $\tau > b$ say $\gamma > (k, u)$ should respect. A natural choice would be that (e.g.) $\beta \supset (0, u)$ should not respect anything, and that $\beta \supset (1, u)$ perhaps might only respect b and perhaps $\beta \supset (2, u)$ should respect for example b^(0, u), b^(0, d) and b^(1, u), b^(1, d) and b^w (see diagram 4)

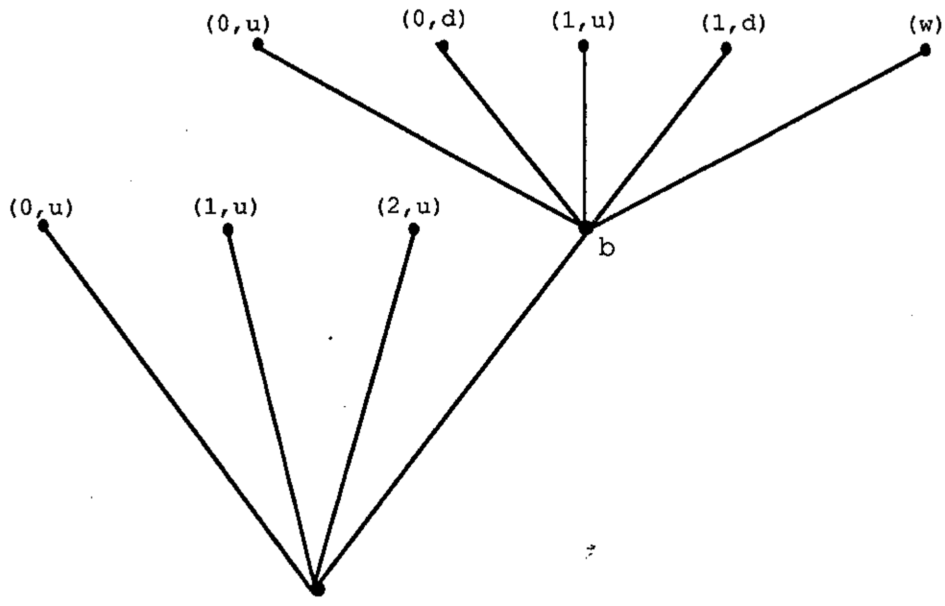


Diagram 4

The idea is that $x_{0,s}$ $x_{1,s}$,and $x_{3,s}$ can only be assigned to such nodes $\tau <^*(2, u)$ and hence if (2, n) appears correct as $x_{3,s}$ appears E-incorrect any restraints associated with it vanish. When we visit (2, u) we only respect this finite set of restraints generated by γ extending (0, u), (1, u) or $\gamma <^*(2, u)$ and $(2, u) \leq_L \gamma$. Note that as a node such as b can have infinitary outcome (e.g. (0, u)) there is a little problem for say $\beta > (2, u)$. In some sense when we visit β it may be that β must respect b's restraint on say $x_{1,s}$.

If we suppose (2, u) is the correct outcome and $x_{1,s} = x(b, 0, s)$ with $0 \notin F$, it may be that the correct b coutcome is (0, u) but when we visit (2, u) b's restraint is up. The point is this. We are - after all - going to have followers apointed for $\gamma \supset (2, u)$. Such followers need to know when it is appropriate for them to enter. This is a familiar enough problem. After all, the potentially bad numbers are only $x_{0,s}$ $x_{1,s}$ and $x_{2,s}$ and we really only need to guess the π_2 behavior of the nodes to which they are assigned.

We can either do this implicitly via the so called "hattrick" approach, or we can be more thematic and expand the tree of outcomes of (2, u). Really we only need to know the *number* of such x_i have infinite activity associated with them, and therefore to guess the π_2 behavior of nodes associated with x_0, x_1, x_2, (2, u) could have a tree of 8 outcomes in diagram 5 below.

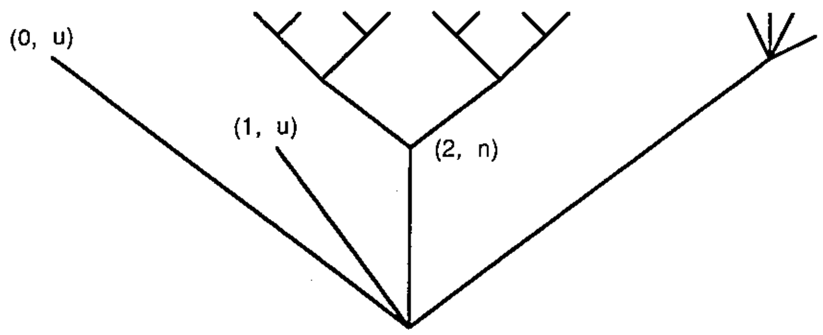


Diagram 5

Doing this expansion would allow us to know exactly the behavior of the nodes associated with x_0, x_1, x_2 if $(2, u) \subset TP$, and this would be the thematic way of performing the construction.

We shall perform a combinatorially simpler approach rather more along the lines of a pinball machine - by noticing we can have only one version of (2, u) if we allow injury from the right. Imagine y is a follower with guess $\gamma \supset (2, u)$ that is F-permitted. We need to add y when the restraints for $\tau <^* \gamma$ become E-incorrect (or at least don't restrain y). The construction will ensure that if $x_{i, s}$ is associated with ρ and $x_{j, s}$ with η and $i \leq j$ then $\rho \leq^* \eta$. The problem is that F needs to sort out if y will enter.

F essentially goes to a stage where $x_{0, s}$, $x_{1, s}$, $x_{2, s}$ cannot later be F-permitted and asks if the computations corresponding to there x_i restraining y are E-correct. If so then y can't enter. The process it must avoid as if (say) $x_{2, s}$ restrains y then later $x_{1, s}$ acts (cancelling the restraint associated with $x_{2, s}$) but it turns out that $x_{1, s}$ has E-incorrect restraint. We note that this can only happen if the node associated ρ with $x_{1, s}$ *initializes* the associated node η with $x_{2, s}$. When this happens we will initialize (2, u) and ensure that, thereafter $x_{2, s}$ can only be associated with nodes $\leq^* \rho$. This device makes the argument and the priority tree such simpler cominatorially and can be used purely because of the way we appoint numbers. (In some other arguments it does not seem to be applicable.) Note that the construction still remains $\leq_T F$.

From the other point of view, nodes $\gamma \supset (2, u)$ will set up restraints whilst they appear satisfied.

Collectively $\{(i, u) : i \in \omega\}$ might restrain all those $\beta \supset b$ from being met. Again we must allow $\tau <^*(2, u)$ priority over the restraints of such γ. This tradeoff ensures that

(2, u) has finite effect on b if b is correct and vice-versa if (2, u) is correct. Hence Harrington's "golden rule" is satisfied (i.e everyone has an environment that coheres with everyone else).

The final problem $\beta \supset b$ might have is that it might not get *stable* followers. For example, take $\eta \supset b$. Now at some b - stage s, we might assign $x_{i,s}$ to η. Now $x_{1,s}$ may be incorrect (as $x_{i-1,s}$ has E-false computations). At the next b - stage t, $x_{i,t}$ ($\neq x_{i,s}$) may be assigned to some γ of lower priority than η (as perhaps t is not a η - stage). Now perhaps γ enumerates $x_{i,t}$ before we visit η. Now $x_{i,t}$ is no longer available for η so we choose $x_{j,t}$ some $j > i$. If this process repeats itself infinitely often then after we lose at η, and we would say η does not get a stable follower.

To overcome this problem, at each stage s for each node γ we have the task of ensuring that there will be infinitely many numbers set aside for γ should γ be visited, and this task is met if $\gamma \subset$ TP. The idea is to simply use a queue. We look back on the construction at the next b - stage t (as above) and see that η was the hightest priority node previously visited with out an inactive follower and see that $x_{i,s}$ was asigned to η. We then declare $x_{i,t}$ as *only to be available for* η (or higher priority nodes) *thereafter* (this is consistent with our earlier devices). Note that activiation only occurs if η is *visited* so the only time infinitely many η - numbers are so "directed" to η is if some $\hat{\eta} \leq \eta$ has the π_2 outcome.

It is clear that the local ordering must be continued throughout the tree and we will give details in (2.16) below. We remark that before we discuss the full construction we will discuss the in (2.18) general machinery of (2.15) in the context of another construction the *Lachlan nonbounding theorem* [10] which is the best known and probably the easiest of the $\mathbf{0}'''$ arguments. If the reader has not altogether followed the previous discussion he or she might like to read this section.

2.16 The Priority Tree and Local Ordering

We shall define the tree T in the obvious way we want "levels" where R_0 occurs at level 0, P_0 at level 1, R_1 at level 2 etc.

The easiest thing to do is to assign a rank for $\gamma \in$ T and define T inductively. Thus λ has rank 0. The outcomes of λ are $\{(i, u) : \tau \in \omega\}$ and b and w. We call λ the *rank 0 node* (written rk(0) = λ).

Now on each of λ's outcomes we will place a copy of P_0 i.e the outcomes

$\{(i, u), (i, d)\ i \in \omega\}$ and w giving each of the outcomes of λ rank 1. Now put a copy of λ's outcomes on each version of P_0's outcomes and these have rank 2 etc.

Similarly we define the local ordering $<^*$ on T inductively in any reasonable way as we did for λ.

This we start λ and for each outcome (j, n) define $\leq^*$ via.

All (i, u) and b are $<^*$ w.

(0, u) : $(0, u) <^* \gamma$ all $\gamma \in T$ and $\gamma \neq (0, u)$ ($\& \gamma \neq \lambda$)

(1, u) : $\tau <^* (1, u)$ for all $\tau \in T$ with $\tau \supset (0, u)$
$b <^* (1, u)$ and $(1, n) <^* \tau$ all $\tau \in \dot{T}$ otherwise.

(2, u) : $\tau <^* (2, u)$ for all $\tau \in T$ with $\tau \supset (1, u)$
$\tau <^* (2, u)$ for all $\tau \in T$ with $\tau <^* (1, u)$ and $\tau <^*$ 02, u)
$\tau <^* (2, u)$ if $\tau = b\hat{}(0, u), b\hat{}(0, d), b\hat{}(1, u), b\hat{}(1, d)$
or $\tau = b\hat{}w$. We have $(2, n) <^* \tau$ otherwise.

In general one extends the ordering above in any reasonable fashion. I am using the idea that (n, u) should be $<^*$ all $\tau <^* (n - 1, u)$ and should respect an increasing finite number of rank $m \leq n$ nodes in such a way as the whole tree to the right of b is eventually enumerated. Also consistency principle is that if σ is a node devoted to P_j then $\sigma\hat{}j <^* \sigma\hat{}i$ iff $\sigma\hat{}j \leq_L \sigma\hat{}i$ and so that the $<^*$ differs only because of even rank nodes.

One can now extend $<^*$ to all of T by an inductive procedure. The only real consistency proviso is that if $\sigma \supset b$ and σ is devoted to R_e then $\tau \supset \sigma\hat{}(i, u)$ with $\tau \supset \sigma\hat{}b$ will agree with the set of $\tau <^* (j, u)$ with $\tau \supset \sigma\hat{}b$ for some j.

2.17 Remark

The reader should think of $<^*$ as the sort of ordering one gets if we were to write the R_e requirements and then constructed the tree with no b - outcome and the (i, u) outcomes (of λ) scattered over the tree. The outcomes (i, u) correspond roughly to a "linking" procedure and the region above (i, u) as a pruned tree constructed via lists as in Slaman/Soare account [16] of there Lachan Nonbounding theorem. We expand on the signifigance of this below (in 2.18).

Theorem (Lachlan [10) $\exists \mathbf{a} \neq \mathbf{0} \; \forall \; (\mathbf{b}, \mathbf{c} \leq \mathbf{a} \; (\mathbf{b} \cap \mathbf{c} = 0 \rightarrow (\mathbf{b} = 0 \vee \mathbf{c} = \mathbf{0}))$

Here one meets the requirements.

$P_e : \bar{A} \neq W_e$

$R_e : \Phi_e (A) = V_e$ and $\Gamma_e (A) = U_e \rightarrow (V_e$ recursive $\vee$ U_e recursive or $((\forall i) (R_{e, i}))$
where
$R_{e, 1} : Q_e \neq \overline{W_i}$.

Here we build $Q_e \leq_T V_e, U_e$ and A and $< \Phi_e, V_e \Gamma_e U_e >_{e \in \omega}$ is an enumeration of all 4-tuples consisting of 2 r.e sets and two functionals. The strategies associated with the above have been discussed in great detail in [10, 15, 16] and we will only give a very brief account (for the sake of completeness).

The basic module for R_e is to have two restraints $r_1 (e, s)$ and $r_2 (e, s)$ we attempt to meet R_e by followers. If we fail one of $V_e \equiv_T \emptyset$ or $U_e \equiv_T \emptyset$ will hold.

The basic module consists of the steps below:-

<u>*Step 1*</u> Pick a follower $x = s$ at stage s and now wait till $l(e, t) > x$ and $x \in W_{i, t}$ for some $t \geq s$, where *for this section* $l(e, s) = \max \{x : (\forall y < x) (\Phi_{e, s} (A_s ; y) = V_{e, s} (y)$ and $\Gamma_{e, s} (A_s ; y) = U_{e, s} (y))\}$.

<u>*Step 2*</u> When t occurs, open a V_e - gap by setting $r_1 (e, t) = 0$, potentially allowing V_e to change.

<u>*Step 3*</u> Wait till the least stage $t_1 > t$ such that $l(e, t_1) > l(e, t)$. Adopt the first case below to pertain.

<u>*Case 3a*</u> (Successful closure) $V_{e,t_1}[x] \neq V_{e,t}[x]$

<u>*Action*</u> Go to 4, setting $r_2(e, t_1) = 0$

<u>*Case 3b*</u> (Unsuccessful closure) $V_{e,t_1}[x] = V_{e,t}[x]$

<u>*Action*</u> Set $r_1 (e, t_1) = t_1$ choose a new follower $x = t$, and go to step 1.

<u>*Step 4*</u> Wait till $t_2 > t_1$ occurs with $l(e, t_2) > l(e, t_1)$ adopt the first case to

pertain

Case 4a (Successful closure) $U_{e,t_2}[x] \neq U_{e,t}[x]$
Action Put x into Q_{e,t_2} meeting $R_{e,i}$. Stop.

Case 4b (Unsuccessful closure) $U_{e,t_2}[x] = U_{e,t}[x]$.
Action Set $r_1(e, t_2) = r_2(e, t_2) = t_2$, pick up a new x and go to step 1.

The outcomes of the module above are, in order of priority f (we reach 4a), g_2 (infintely many case 4b), g_1(infintely many 3b, but only finitely many 4b) and w (stay waiting). To use our set up, a node σ on the tree devoted to R_e would have $\omega + 2$ outcomes labelled

$$(0, g_2), 0, g_1), (1, g_2), (1, g_1), \_\_,\_, f, w.$$

where (i, g_j) denotes the outcome that $R_{e,i}$ has outcome g_j, f is the π_3 outcome : that all the $R_{e,i}$ have finite behaviour and w is $l(e, w) \not\to \infty$ outcome.

Again we would define a local ordering $<^*$ on the tree made up from such primary outcomes where outcome (i, g_1) would be refined by having a finite tree of suboutcomes corresponding to the π_2 behaviour of those $\tau <^* (i, g_j)$ with $\tau \supset s$^ f(this is easier the most general such theorem as we don't need the whole tree only the number of nodes that exhibit π_2 behaviour).

We play outcome f whenever it looks correct. here that would mean that we would have a monotone maker m(e, s) and, whenever we see that $R_{e,0}, \_\_ R_{e,m(e,s)}$ all exhibit finite behaviour (i.e. either waiting for $R_{e,i}$'s current x to enter $W_{i,s}$ or we get the 4a for $R_{e,i}$) we play f and set m(e, s + 1) = m(e, s) + 1.

Note that if (i, g_j) is the correct primary outcome of R_e at σ then those $\gamma \leq_L \sigma$^(i, g_j) only have finite effect, and there are only finitely many $\beta <^* \sigma$ with $\beta \leq_L \sigma$. These exhibit at worşt π_2 beaviour and this beaviour can be guessed as the correct sub -outcome μ of σ^(k, g_j). This means that the restraint $r_j(\sigma$^(i, g_j)^ μ, s) holdsV_e (j = 1) or U_e (j = 2) during the co-gaps and there is no change in the gaps. It follows that if e.g. j = 1, then V_e is recursive.

We remark that we feel in some sense that this technique is to **0'''** - arguments is what the tree method was to **0''** - arguments, as in some sense, it is the 'natural' method of doing such arguments. We shall expand on these comments later in §3.

2.18 Notes on the Nonbounding Theorem, the General Machinery and the 0''' Method

As we remarked in (2.17) the $<^*$ priority ordering can be thought of as the alternative method for the construction. (We also remark that one can use a more dynamic notion of local ordering. This device was used by Shore[13] and a forth coming paper of Downey and Shore (see the note at end of paper).) The module for Slaman's density theorem is, itself, quite complicated and the fact that the construction remains $\leq_T$ F quite delicate. To aide the reader will take time out to discuss the machinery of (2.15) and (2.16) in the context of a much better know (and easier construction). Here we recall the nonbounding theorem of Lachan.

2.19 The Construction and Verification

The inductive strategies in this result are the same as those for the α-module described earlier. The basic rules are that if y is associated with $\sigma \in T$ then $y < |\sigma|$. If $\gamma \supset \sigma$ and $x_{i,s}$ is a member of σ's stream at s and associated with γ then for all stages $t \geq s$, $x_{i,s}$ is associated with only η such that $\eta \leq^* \gamma$, and if $x_{i,s}$ is associated with γ and some $\eta <^* \gamma$ asserts control of $x_{i,s}$ at stage s (and declares $x_{i,s}$ to no longer follow η (e.g. $x_{i,s} < r(n, s)$) then for all $t \geq s$, $x_{i,t}$ can only be associated with $\rho \leq^* \eta$.

If σ is associated with R_e and we see some $x_{i,s}$ e-computation have E-incorrect use, we enumerate $x_{i+1,s}$ as described in the α-module. Namely $x_{i+1,s}$ enters if nobody controls $x_{i+1,s}$ or if both sides change, and $x_{i+1,s}$ does not enter if only one side changes but the other side has higher priority control asserted on it. We will play a $\sigma^\wedge(i, u)$ stage the next time we visit σ. If r (e, s) is preserving or measuring a computation at ρ and this is E-incorrect at s + 1 we cancel the restraint. If $x_{i,s} = x(\gamma, j, s)$ ($|\gamma|$ odd) is active and j occurs in F at s, we declare $x_{i,s}$ as F-permitted at γ. This remains so for all stages $t \geq s$ unless some higher priority $x_{j,s}$ acts, the region over which γ assert control is E-incorrect.

The construction then proceeds in the obvious way in substages. Starting at λ we see find the highest priority option amongs $\sigma^\wedge(i, u)$ appears correct, $\sigma^\wedge b$ appears correct, $\sigma^\wedge w$ appears correct, or some τ requires attention (that is some $x(\tau, j, s)$ which is F-permitted is, (now) unrestrained).

One then chooses the relevant option and this gives $\sigma_0 = \lambda$, and $\sigma_1 \supset \sigma_0$. We proceed until $|\sigma_s| = s$ inductively.

The verification of the construction is virtually the same as the earlier discussion and we will this omit it, and let this conclude our notes on Slaman's density theorem.

§3 EXTENSIONS, VARIATIONS AND OTHER RESULTS USING THE TECHNIQUE

The proof of Slaman's density thereom, as well as the general machinery of §2 admits several further extensions and applications: First, a miner modification to the P_j will give an embedding of the countable atomless boolean algebra into any $[\mathbf{e}, \mathbf{f}]$ with $\mathbf{e} < \mathbf{f}$. This follows by virtually the same argument (see e.g. [3]).

Actually we feel that by varying especially the Friedberg type requirements P_j one can obtain many other density results. One example is the density of the *superbranching* degrees. Here a degree $\mathbf{a} \neq \mathbf{0'}$ is superbranching (Downey - Mourad [4]) if for all $\mathbf{b} > \mathbf{a}$ there exist $\mathbf{c}, \mathbf{d}$ such that $\mathbf{a} < \mathbf{c}, \mathbf{d} < \mathbf{b}$ and $\mathbf{c} \cap \mathbf{d} = \mathbf{a}$.

To see this we need to consider how one might make such an $\mathbf{a}$. We would build $A = \bigcup_s A_s$,

$C_e = \bigcup_s C_{e,s}$ and $D_e = \bigcup_s D_{e,s}$ to meet
$P_{2<e,i>} : (W_e \leq_T A) \vee (\Phi_i(A) \neq C_e)$
$P_{2<e,i>+1} : (W_e \leq_T A) \vee (\Phi_i(A) \neq D_e)$ with $C_e, D_e \leq_T W_e \oplus A$, and
$N_{e,i} : \Phi_e(\hat{C}_e) = \Phi_e(\hat{D}_e) = t$ total $\Rightarrow t \leq_T A$ where $\hat{C}_e = C_e \oplus A$ and $\hat{D}_e = D_e \oplus A$.

Let $l(e, i, s) = \mathrm{mx}\{x : (\forall y < x)(\Phi_{e,s}(C_{i,s}; y) = \Phi_{e,s}(D_{i,s}; y))\}$ and $ml(e, i, s) = \mathrm{mx}\{l(e, i, t) : t < s\}$.

To meet the $P_{2(e,i)}$ we will define a stream of followers $\{x(e, i, j, s) : j \in \omega\}$- we shall wait until $L(e, i, s) > x(e, i, j, s)$ where

$L(e, i, s) = \max\{x : (\forall y < x)[\Phi_{i,s}(A_s; y) = C_{e,s}(y)]\}$, and then appoint $x(e, i, j + 1, s) > u(\Phi_{i,s}(A_s; x(e, i, j, s)))$ in the same way as (2.3). Again we shall wait until we see W_e permit some j and then put $x(e, i, j, s)$ into C_e.

This wins $P_{2(e,i)}$ by a Friedberg argument if we restrain A on (e.g.) $x(e, i, j + 1, s)$. As we don't know whether $W_e \leq_T A$, we may get all $x(e, i, j, s)$ defined for $j \in \omega$ (perhaps $W_e = \emptyset$). Subsequent P_k must this take numbers from $P_{2<e,i>}$'s stream but this causes no new problems save to ensure that $\lim_s x(e, i, j, s)$ exists, this being simply done by controlling which P_k have access to which $x(e, i, j, s)$ (see [4] for more details).

The $N_{e, i}$ are met in exactly the same way as we did in (2.2). The only problems with coherence with density are those that have occurred in the last section and virtually the same reasoning shows that:

3.1 Corollary. The Superbranching Degrees are Dense

It seems to me that the same technique will also show that many other degree classes are dense. For example the same technique ought to show that the contiguous degrees are nowhere dense in **R**, that is, if $\mathbf{e} < \mathbf{f}$ then there exists $\mathbf{a} < \mathbf{b}$ with $\mathbf{a} < \mathbf{a} < \mathbf{b} < \mathbf{f}$ such that for all $\mathbf{c} \in [\mathbf{a}, \mathbf{b}]$, $\mathbf{c}$ is not contiguous.

As a final example of the use of this technique, we will sketch the proof of the following result which negatively answers a conjective of Remmel.

3.2 ***Theorem*** *There exists an re degree* $\mathbf{a}$ *with* $\mathbf{0} < \mathbf{a} < \mathbf{0'}$ *such that if* B *is any* r.e *set with* $\deg(B) > \mathbf{a}$ *then there exists an r.e splitting* $B_1 \sqcup B_2 = B$ *of* B *with* $\deg(B_1) = \mathbf{a}$.

Proof Let $E \not\equiv_T \emptyset$ be given. We construct $A = \bigcup_s A_s$ and auxiliary re. set E_e, C_e and D_e meet

$$P_e : \bar{A} \neq W_e$$
$$N_e : \Phi_e(A) \neq E, \text{ and}$$
$$R_e : \Gamma_e(V_e) = A \to (C_e \sqcup D_e = V_e \text{ and } C_e \equiv_T A.).$$

Here (Γ_e, V_e) is an enumeration of all pairs consisting of an r.e. set and a functional. We meet the P_e by followers as usual. We shall meet the N_e by a Sacks restraint and shall only have problems with the R_e. For the sake of there we shall define reductions (dropping the "e") $\Delta(C) = A$ and $\Psi(A) = C$.

The basic idea for R is to wait till
$l(e, s) > x$ (where $l(e, s) = \max\{x : (\forall y < x)(\Gamma_{e,s}(V_{e,s}; y) = A_s(y)\}$ and define
$\delta(x, s) = \gamma(x, s) = u(\Gamma_{e,s}(V_{e,s}; x))$ and $\psi(\delta(x, s)) = \langle e + 1, x, s\rangle$.

Were there no N_e requirements around, the idea is then quite simple: Whenever

$V_{e,s}[\delta(x, s)] \neq V_{e,t}[\delta(x, s)]$ at some least e-expansionary stage $t > s$ enumerate the change into $C_{t+1} - C_t$ and $\psi(\delta(x, s))$ into $A_{t+1} - A_t$ and then choose a new $\psi(\delta(x, t+1))$ as, say, $<e+1, x, t+1>$ and again set $\delta(x, t+1) = \gamma(x, t+1)$. Note that if we use $y = <0, x, s>$ to follow P_e then if we ever put such y into A it will cause a change in $\gamma(y, s)$ and hence in $\delta(x, s)$. Note also that if $\Gamma(V) = A$ then $\lim_s \delta(x, s) = \delta(x)$ and $\lim_s \psi(\delta(x, s)) = \psi(\delta(x))$ exists, and further as the use functions are, by convention monotone and $\delta(x) \geq x$, we will have a reduction $\Psi(A) = C$.

Unfortunately the procedure above makes A = C and complete. However, the reader should note that R_e has already $\omega + 2$ outcomes, that is outcome (i, u) indicating i is least with $\Gamma(V, i)$ unbounded (for $i \in \omega$) plus b and w as in §2. Note that outcome (i, u) essentially codes a recursive set into A provided we ensure that $\psi(\delta(x)) > \psi(\delta(y))$ when $x > y$. This follows because we get to reset all traces $\psi(\delta(x, s))$ if $\delta(x, s)$ changes.

N_e wishes to stop A from changing. It asserts control of $\psi(\delta(x, s))$ (say) when it sees $L(e, s) > y$ $\left(\text{where } L(i, s) = \text{mx} \left\{x : (\forall y < x)(\Phi_{e,s}(A_s ; y) = E_s(y))\right\}\right)$ and $\psi = \psi(\delta(x, s)) < u(i, y, s) = u(\Phi_{i,s}(A_s ; y))$. This control asks us to keep ψ out of A.

This causes problems. As V is not under our control $V[\delta(x, s)]$ might change after N_i asserts control. If we do not enumerate ψ our only option is to enumerate the relevant changes into D (and not C). But note now that x is finished as a possible follower in the sense that as $\delta(x, s)$ is now no longer equal to $\gamma(x, s)$, if x enters A it may not cause a change in V below $\delta(x, s)$. Hence we may not be able to get C to comprehend that such an x enters.

The machinery of (2.15) and (2.16) handles this situation very nicely. We put the R_e, P_e and N_e on a tree in the same way as in §2 defining a local priority ordering $<^*$. Then if τ corresponds to an N_e and σ to R_e with $\tau \supset \sigma$^b then σ^(i, u) respect τ's control precisely if $\tau <^* \sigma$^(i, u). This restraint again has finite lim inf (and so has finite\limit on the true path on the expanded tree for σ^(i, u)) and so if $\Gamma_e(V_e ; i)\uparrow$ then almost all of the $\psi_e(\delta_e(x, s))$ enter A (i.e. all those $\geq$ j some $j \geq i$). Those N_e guessing σ^(i, u) will not believe computations until this recursive set enters. On the other hand, for $\tau \supset \sigma$^b then if some $\gamma \supset \sigma$^b higher priority than τ is devoted to D_e, it will request an x to follow it. It will wait until it sees a fresh x provided by σ^b. It asserts control of x by asking that $\psi(x)$ and $\delta(x)$ move everytime $\gamma(x)$ changes.

The remaining details fit together in exactly the same way as the other arguments and this concludes the sketch of the proof of this result.

3.3 *Final Notes on the Framework*

Finally, we point out a couple of remarks on the framework. We feel that the beauty of the approach is that for a general $\mathbf{0}'''$ argument are simply writes down a π_3 requirement and then give a module for *the whole requirement exactly as in a* $\mathbf{0}''$ *argument*. The $<^*$ machinery then ought to take care of the coherence problems (in the same way as the $\leq_L$ tree machinery takes care of $\mathbf{0}''$ arguments).

The disadvantage of our approach is that the combinatorics seem more difficult, than, for example, the linking of Slaman/Soare (at least for some arguments).

This would give a simultaneous extension of Slaman's density theorem and Lachlan's decomposition theorem. It appears to be quite difficult.

Richard Shore has pointed out that one can also construct the local priority ordering during the construction. For example, if the ω π_2 outcomes are labelled i and the π_3 outcome is labelled b then the number of times i is accessed can control the number nodes $\sigma \supset b$ that $i + 1$ must respect. Note that if b is correct then this number will be finite. Similar comments apply from σ to i. Shore used this device in his construction; and it can make the combinatorics easier. Downey and Shore have employed this technique in a forthcoming paper entitled "Decomposition and infima in the r.e. degrees". There Downey and Shore obtain the following generalization of Slaman's density theorem: $\forall\, \mathbf{a}, \mathbf{b}\, (\mathbf{a} \mid \mathbf{b} \rightarrow (\exists\, \mathbf{c})$ $(\mathbf{a} \cup \mathbf{c} \mid \mathbf{b} \cup \mathbf{c}$ and $\mathbf{a} \cup \mathbf{c}, \mathbf{b} \cup \mathbf{c} < \mathbf{a} \cup \mathbf{b}$ and $(\mathbf{a} \cup \mathbf{c}) \cap (\mathbf{b} \cup \mathbf{c}) = \mathbf{c}))$. Finally Lerman has pointed out that the use of π_3 trees also occurs in his paper on degrees that do not bound minimal degrees.

REFERENCES

1. Ambos - Spies K., On pairs of recursively enumerable degrees Trans. Amer. Math. Soc. *283* (1984) 507-531.

2. Downey, R. G., A contiguous nonbranching degree, Z. Math. Logik. Grundlagen Math *35* (1989) 375-383.

3. _________, Lattice nonembeddings and initial segments of the r.e. degrees. Annals Pure and Aplied Logic (to appear)

4. _________, and J Mourad, Superbranching degrees, these proceedings

5. _________, and T Slaman, Completely mitotic r.e. degrees, Annals Pure and Appl. Logic *41* (1989) 119-152.

6. _________, and T Slaman, On co-simple isols and their intersection types, in preparation.

7. Fejer, P., *The Structure of Definable Subclasses of the Recursively Enumerable Degrees,* Ph. D. Diss., Univ. of Chicago, 1980.

8. _________, The density of the nonbranching degrees, Annals Pure and Appl. Logic *24* (1983) 113-130.

9. Lachlan, A. H., A recursively enumerable degree which will not split over all lesser ones, Ann. Math. Logic *9* (1975) 307-365.

10. _________, Bounding minimal pairs, J. Symb. Logic *44* (1979) 626-642.

11. _________, Decomposition of recursively enumerable degrees, Proc. Amer. Math. Soc. *79* (1980) 629-634.

12. Sacks, G. E., The recursively enumerable degrees are dense, Ann. of Math *80* (1964) 300-312.

13. Shore, R. A., A non-inversion theorem for the jump operator, Ann. Pure and Appl. Logic *40* (1988) 277-303.

14. Slaman, T. A., The recursively enumerable branching degrees are dense in the recursively enumerable degrees, handwritten notes, University of Chicago, 1981.

15. Soare, R. I., Tree arguments in recursion theory and the $\mathbf{0}'''$ priority method, in *Recursion Theory* (ed. A. Nerode and R. Shore) A.M.S. publ. Providence, Rhode Island (1985) 53-106.

16. _________, *Recursively Enumerable Sets and Degrees,* Springer-Verlag, New York (1987).