Notes on Higher Type ITTM-recursion

P.D. Welch [Draft notes, not for circulation please]

5 Aug 2021

The purpose of these notes is to outline a theory of higher type recursion (actually just type 2) for infinite time Turing machines (ittm's) in the manner of Kleene from the late 1950's and early 1960's, in particular where he used (ordinary) Turing machines arranged on wellfounded trees to present computations and their subcomputations (see [9],[10]). Since the advent of the ittm model in [5] it was perhaps clear that something like this could be done, but without anyone putting the shoulder to the wheel. (What I do *not* mean here, is the computation on elements of Cantor space, with an oracle some $A \subseteq 2^{\mathbb{N}}$ that was also outlined in [5]. Although, to state the obvious, ittm's allow for computation using reals (identified with elements of Cantor space), that simple recursive model does not have the Kleenean feature of calling subcomputations, which enabled him to build up a model of recursion using Turing machines that, for example showed that the Kleene-decidable sets coincided with the Δ^1_1: we want to also build a model that equally goes beyond the basic oracle machine.)

We outline a theory of this *generalised type-2 ittm recursion* here. A number of choices were made as to what features the model should have, and no doubt there are variations. A typical oracle in this theory is then a total type-2 functional I: ${}^k\omega \times {}^l({}^\omega 2) \to \omega$ - just as for Kleene type-2 recursion, and we develop the theory of recursive-in-I functions $\{e\}^{\mathsf{I}}$: ${}^k\omega \times {}^l({}^\omega 2) \to \omega$ as exemplified by a machine architecture $P^{\mathsf{I}}_e(\boldsymbol{m}, \boldsymbol{x})$ with e thought of as an index of a program - as always a finite ordinary Turing programme, but enhanced with a query call instruction initiating subcomputations as Kleene did.

Once the model is defined, the question of its properties arises. For Kleene recursion the existential quantifier ${}^2\mathsf{E}$ played a crucial role, and the theory was developed often with the stipulation that 'normal' fumctionals should be candidates for oracles, *i.e.* that an oracle I was normal, if ${}^2\mathsf{E}$ was itself Kleene recursive in I. We use $\mathsf{E} = {}^2\mathsf{E}$ here also as a 'base' oracle for our arguments, although here its role is rather trivial. Any 'suitable' functional (as defined below) would do.

One immediately wants to ask: what are the semi-decidable sets? What is the 'E-halting problem'?

$$H = \{e | \{e\}^{\mathsf{E}}(e)\downarrow\}.$$

The analogue of $\omega_1^{\mathrm{ck}} = \omega_1^{\mathsf{E},\mathrm{ck}}$ - the least ordinal not the order type of a (Kleene) recursive (in E) wellordering of ω, is here α^{E}_0, the least ordinal not the order type of a wellordering of ω which is generalised ittm recursive in E. It turns out that this is a rather large countable ordinal.

Infinite Time Turing Machine theory has developed since its inception in [5] and there have been many ingenious ways of expanding its role to define 'computable' concepts. It was clear from [5] that the model could produce codes for some initial segment of the L_α-hierarchy. One generalisation was due to Peter Koepke and simply extends the tape to have length On, [12]. This produces codes for any L_α - given a mark for the ordinal α on its tape. Thus a satisfaction relation for $(L, \in)$ could be so computable, whilst the original ITTM's were limited to doing this for an initial segment of true ω_1. There had been the hope that such devices could give, when not new proofs of fine structural results of Jensen, such as say the $\square$ principle, at least new insight into the fine-structure of L. However this seems not really to be the case: the Σ_2-nature of the liminf rule used for cell update at limit stages works against the very Σ_1, or even Σ_0 method of skolem hulls used in fine-structural proofs. So although α-recursion theory can be somewhat reformulated as α-length tape ittm models no new deeper fine-structural insights seemed to be forthcoming.

However the author has for some while had the thought that the strength of ittm theory was either exactly, or at least close to, being lined up with that of Σ^0_3-Determinacy. (The reasons for this insight are a little difficult to state just here.) If this were to be so, it would be a characterisation of this classical descriptive set theoretic property by means of this generalised machine theory: it would be the occasion of a true application of ittm-theory to classical DST, and as far as we are aware, it would be the first such.

This can be realised as follows:

Theorem 1. *Let U be a complete $\Im\Sigma^0_3$-set. Then $H \equiv_1 U$, that is, the halting set is recursively isomorphic to a complete $\Im\Sigma^0_3$-set of integers.*

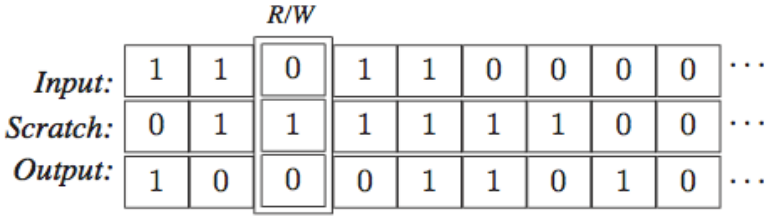
That is to say, if the Σ^0_3 sets of reals are (ordinarily) recursively listed as say $A_0, A_1,..., A_n, ...$, then there is a pencil and paper algorithm f given an element $k \in H$ to calculate an $f(k)$ so that Player I has a winning strategy in $A_{f(k)}$ *and vice versa*. That is, $f\colon \omega \longrightarrow \omega$ is an ordinary Turing computable bijection and we thus have, given H, a complete listing of those games in which I wins, and conversely from such a listing we can retrieve H using the inverse of f. Thus H and a $\Im\Sigma^0_3$-set are (ordinarily) *recursively isomorphic*. A further analogy with ω_1^{ck} and Σ^0_1-Determinacy emerges: just as strategies for Player I in Σ^0_1-games appear all the way up to stage $L_{\omega_1^{ck}}$ in the L-hierarchy, so strategies for Player I in Σ^0_3-games appear all the way up to stage $L_{\alpha_0^{\mathsf{E}}}$, thus giving a second characterisation of what α_0^{E} "is".

Theorem 2. *If A is a $\Sigma^0_3(x)$ set so that the game $G(A)$ is won by Player I, then there is a generalised-ittm-recursively computable (in x) winning strategy σ for I. That is, for some index e dependent on the definition of A, but not x, $\{e\}^{\mathsf{E}}(x)\downarrow\sigma$.*

The notes that follow are intended to be an introduction to this generalised type-2 ittm recursion.

1 Infinite Time Turing Machines

1.1 Basic Construction



The Turing programmes are enumerated as $\langle P_e |\ e \in \omega\rangle$ with 'indices' e. We suppose the programme is listed as a finite list of instructions I_0, ..., I_k. The *current instruction number* at time $\alpha+1$, $I(\alpha+1)$, and its R/W action *etc.* is determined by the programme in the usual Turing way.

At limit times λ we use Liminf behaviour. If the cells of the machine are enumerated $\langle C_i |\ i \in \omega\rangle$ with values at time ν denoted by $\langle C_i(\nu) |\ i \in \omega\rangle$ then we set at limit time λ:

$$C_i(\lambda) = 1 \Longleftrightarrow \exists\alpha < \lambda \forall \beta < \lambda (\alpha < \beta \longrightarrow C_i(\beta) = 1)\ ;$$

Thus, if the value if the $C_i(\alpha)$ alternates 0/1 cofinally often in some limit ordinal λ then $C_i(\lambda) = 0$. The *current instruction number* $I(\lambda)$ at a limit time λ is $\mathrm{Liminf}_{\alpha \longrightarrow \lambda} I(\alpha)$. "OT" abbreviatse "output tape".

Definition 1.1. *We say* $P_e(x)$ *is* convergent *and write*

$$P_e(x)\downarrow \text{ if } \exists \tau (\text{the OT of } P_e(x) \text{ remains constant after time } \tau).$$

If the contents of that OT is the real y *we write* $P_e(x)\downarrow y$. *If we wish to indicate the time* τ *at which this occurs, we write* $P_e(x)\downarrow^{\tau} y$. *It is* divergent *otherwise, and we write* $P_e(x)\uparrow$.

A synonym for convergence is being "eventually settled". The machine may not formerly halt, but it makes no further changes to the OT. We treat formal machine halting as a special kind of convergence.

Definition 1.2. *The* snapshot at time or stage α is $s_\alpha = s_\alpha(e,x)$ *of* $P_e(x)$ *is a sequence consisting of the current instruction number* $I(\alpha)$ *about to be performed, an integer* $R(\alpha)$ *representing the position of the R/W head, and an* ω*-sequence of* $0/1$ *representing the cell contents at time* α: $\langle C_k(\alpha) | k < \omega \rangle$.

We note the following:

Observation: a course of computation of some $P_e(x)$ with input $x \in 2^{\mathbb{N}}$ is absolute to $L[x]$ and can be defined by a Δ_1-recursion so that $\langle s_\beta |\ \beta < \alpha \rangle \in \Delta_1^{J_\alpha[x]}$ with the snapshot s_α at stage α being $\Sigma_2^{J_\alpha[x]}$.

Definition 1.3. *An ordinal* ξ *is called* Σ_2-extendible *if there is* $\delta > \xi$ *with* $L_\xi \prec_{\Sigma_2} L_\delta$. *We call* (ξ, δ) *a* (Σ_2)*-extendible pair. The least such pair is usually written* (ζ, Σ).

Remark: For the least pair (ζ, Σ), L_Σ is the unique such end extension. Such a ξ is always a Σ_2-admissible, and is a limit of such; δ is also a limit of such, but need not be itself $(\Sigma_1$-)admissible.

Exercise 1.1. Verify all of the last Remark. [Hint: Argue that Σ_2-Collection holding in L_ξ follows from its Σ_2-extendibility. To show that, *e.g.*, Σ is not admissible, consider $T^2_\alpha =_{\mathrm{df}} \Sigma_2\text{-Th}(L_\alpha) \in L_{\alpha+1}$. By extendibility, $T =_{\mathrm{df}} T^2_\zeta = T^2_\Sigma$. Let $\alpha_0 = \zeta$ and $\alpha_n > \alpha_{n-1}$ be least with $T \cap n = T^2_{\alpha_n} \cap n$ (where we think of sentences as recursively coded by integers).]

Remark: The notion easily relativises: we say that ξ is $\boldsymbol{x}$*-extendible* if there is $\delta > \xi$ with $L_\xi[\boldsymbol{x}] \prec_{\Sigma_2} L_\delta[\boldsymbol{x}]$ and so forth.

By the Observation we have:

Lemma 1.4. $P_e(n)$ *has identical snapshots at times* ζ, Σ: $s_\zeta = s_\Sigma$.

Having identical snapshots is close to be being forever looping.

Definition 1.5. *We shall say that a computation such as* $P_e(x)$ 'exhibits final looping behaviour' *(*'at stage σ', *or* 'by stage τ'*), if there are stages or times* $\xi < \sigma (\leq \tau)$ *with (a) identical snapshots at* ξ *and* σ*, and moreover (b) no cell that had a stable value at time* ξ *changes that value in the interval* (ξ, σ). *We say the computation* 'has entered a repeating loop' *at time or stage* α *if* $\xi \leq \alpha$ *for such a pair* (ξ, σ).

1.2 The Theory Machine TM

We sketch a presentation of the "Theory Machine" of [2]. We assume familiarity with the J_α-hierarchy (those not so familiar with this can pretty much read "L_α" for "J_α" throughout with little distortion of the truth; recall that in any case if $\alpha = \omega \cdot \alpha$ then $L_\alpha = J_\alpha$).

Definition 1.6. *Let* $S^n_\alpha =_{\rm df} \{\beta < \alpha |\ J_\beta \prec_{\Sigma_n} J_\alpha\} \cup \{0\}$.

Note 1.7. (i) A $\beta \in S^1_\alpha$ we shall say is "Σ_1-*stable in* α".

(ii) If S^1_α is unbounded in α we say that α is non-Σ_1-projectible, and in fine structural terminology "$\rho^1_\alpha = \alpha$". One may reason that $S^1_\alpha \in \Pi^{J_\alpha}_1$.

(iii) We use the observation that if σ is a Σ_2-sentence, then for $\beta \in S^1_\alpha$, that $J_\beta \vDash \sigma \Rightarrow J_\alpha \vDash \sigma$ (since if $\sigma \equiv \exists u \psi(u)$ with $\psi \in \Pi_1$, if $u_0 \in J_\beta \vDash \psi[u_0]$ then by upwards persistence of Π_1 sentences $J_\alpha \vDash \psi[u_0]$).

(iv) We recall the existence of a uniform (in α) Σ_1-definable skolem function h^1, so that when h^1_α is this function defined in J_α then for any $A \subseteq J_\alpha$, then h^1_α"$\omega \times (A \cup \{\varnothing\})$ is the least Σ_1-skolem hull containing A in J_α. We write often h_α for h^1_α. Thus h_α"$\omega \times A \prec_{\Sigma_1} J_\alpha$. In particular if $\alpha < \alpha'$ then $h_\alpha \subseteq h_{\alpha'}$.

(v) We also have that if $\beta = \max S^1_\alpha$ then (a) h^1_α"$\omega \times \beta + 1 = J_\alpha$; and (b) if $J_\alpha \vDash$ "β is countable" then h^1_α"$\omega \times \{\beta\} = J_\alpha$. ((b) follows from (a), since then there is an onto function $f \in J_\alpha$, f: $\omega \longrightarrow \beta$, with f Σ_1 definable in J_α from the parameter β; *i.e.* $f \in h^1_\alpha$"$\omega \times \{\beta\}$).

Definition 1.8. *(i) Let* $T^n_\alpha =_{\rm df} \{\ulcorner \sigma \urcorner \in \omega |\ \sigma$ *is* $\Sigma_2 \wedge J_\alpha \vDash \sigma\}$ *;*
(ii) $\widehat{T_\alpha} =_{\rm df} \{\ulcorner \sigma \urcorner \in \omega |\ \ \sigma$ *is* $\Sigma_2 \wedge\ \exists \beta < \alpha \forall \tau \in (\beta, \alpha)\ \sigma \in T^2_\tau\}$.

Then $\widehat{T_\alpha}$ is the set of Σ_2-sentences that are 'eventually true' below α. Two preliminary lemmas are needed before describing the ittm theory machine.

Lemma 1.9. *There is an (ordinary) Turing recursive function* $f : \omega \times \omega \longrightarrow \omega$*, so given by an index* e*, so that for any* $\mathrm{Lim}(\lambda)$ *satisfying* $J_\lambda \vDash$ *"Every set* x *is countable", if we set* $T = \widehat{T_\lambda}$ *then* T^2_λ *is uniformly r.e. in* T*, via* f*, that is* $\ulcorner \sigma \urcorner \in T^2_\lambda \longleftrightarrow \exists i\, f(i, \ulcorner \sigma \urcorner) \in T$.

Proof: Let $\sigma \equiv \exists u \psi(u)$ with $\psi \in \Pi_1$. We
Claim: $\sigma \in T^2_\lambda \longleftrightarrow \exists i [\exists \tau_0 \forall \tau \in (\tau_0, \lambda)$
$J_\tau \vDash$ "$\exists \beta \in S^1_\tau \big((\sigma)^{J_\beta} \vee (h_\tau(i, \beta)\downarrow \wedge \psi[h_\tau(i, \beta)]^{J_\tau}) \big)$"$]$

Note first that the expression in quotation marks on the right hand side, $\eta_\sigma(i)$ say, here is, if true, a member of T^2_τ, being $\Sigma^{J_\tau}_2$ in i. We thus shall have $\sigma \in T^2_\lambda \leftrightarrow \exists i \ulcorner \eta_\sigma(i) \urcorner \in T$ and the Lemma is proven.

Proof: of Claim.

Case 1. S^1_λ is unbounded in λ.

Suppose the left hand side holds of σ. Suppose $\psi(u_0)^{J_\lambda}$ holds for u_0.Then for some sufficiently large $\beta \in S^1_\lambda$, $u_0 \in J_\beta$, and then $\psi(u_0)^{J_\beta}$. But $\beta \in S^1_\lambda \longrightarrow \beta \in S^1_\tau$ for any $\tau > \beta$; consequently the first disjunct of the right hand side holds. For the converse direction, fix the given i. By the Case hypothesis we can assume that τ itself is in S^1_λ. But then if the first disjunct holds, if $(\sigma)^{J_\beta}$ and $\beta \in S^1_\tau$ then $\beta \in S^1_\lambda$ and thence $(\sigma)^{J_\lambda}$. If the second disjunct holds for the supposed i $\psi[h_\tau(i, \beta)]^{J_\lambda}$ holds for the same reasons.

Case 2 $\beta_0 =_{\rm df} \max S^1_\lambda < \lambda$ exists.

By the bullet points above every $x \in J_\lambda$ is of the form $h_\lambda(i, \beta_0)$. Again suppose the left hand side holds of σ and $\psi(u_0)^{J_\lambda}$ holds for u_0. In particular now $u_0 = h_\lambda(i, \beta_0)$ for some i. Let $\tau_0 \geq \beta_0$ be sufficiently large so that $(h_{\tau_0}(i, \beta)\downarrow)^{J_{\tau_0}}$ and thence, by the fact of ψ being Π_1, $(\psi[h_{\tau_0}(i, \beta)])^{J_{\tau_0}}$. By the upwards persistence of Σ_1 formulae in the first case and downwards persisitence of ψ in the second case, these will hold in all larger J_τ for $\tau \leq \lambda$ replacing τ_0. But now the second disjunct of the right hand side holds.

Conversely suppose the right hand side holds. Let i be as supposed. By the maximality of β_0 for unboundedly many $\tau' \in (\beta_0, \lambda)$ some new Σ_1-sentence about β_0 becomes true first in $J_{\tau'+1}$. Pick such a $\tau = \tau' + 1$ of this form. Such a τ ensures that $S^1_\tau = S^1_\lambda$ and thence $\max S^1_\tau = \beta_0$ too. So suppose the first disjunct holds for such a successor τ. Then if $(\sigma)^{J_\beta}$ holds for a $\beta \in S^1_\tau = S^1_\lambda$ we shall have $(\sigma)^{J_\lambda}$ and we are done. Thus we now suppose the first disjunct fails for τ of this form; pick any such τ, then the second disjunct holds as witnessed by a $\beta \in S^1_\tau$.

Then if $\beta < \beta_0$ then $(\exists y(y = h_\tau(i, \beta)))^{J_\tau}$ implies $(\exists y(y = h_{\beta_0}(i, \beta)))^{J_{\beta_0}}$ by the uniformity of the definition of the Σ_1-skolem function h, and the fact of $\beta_0 \in S^1_\tau$. But then $\big(h_{\beta_0}(i, \beta)\downarrow \wedge \psi[h_{\beta_0}(i, \beta)]^{J_{\beta_0}}\big)$. But this entails that the first conjunct holds for τ, which we are assuming does not happen. Hence we must have $\beta = \beta_0$. However then $\psi[h_\tau(i, \beta_0)]^{J_\tau}$ for any τ of this form, and so for such τ arbitrarily large below λ. By the upwards persistence of $h_\gamma(i, \beta_0)$ for $\gamma \in [\tau, \lambda]$ we have $\psi[h_\tau(i, \beta_0)]^{J_\lambda}$ and hence σ^{J_λ}. QED

Lemma 1.10. *Let ζ be the least Σ_2-extendible ordinal and let Σ be its extension: $L_\zeta \prec_{\Sigma_2} L_\Sigma$.*

(i) There is a uniform procedure ittm-recursive in T^2_α in ω steps, for any $\alpha < \Sigma$, for computing a real x_α which is a code for the structure $\langle J_\alpha, \in\rangle$.

(ii) There is thus a partial onto function $f \in \Sigma_2^{J_\alpha}$, $f : \omega \twoheadrightarrow J_\alpha$.

("Uniform" here means the procedure is independent of α.) See [2].

Proof: (Sketch) (i) It suffices to note that there is a uniform Σ_2-skolem function $h = h^2_\alpha$ with domain a subset of $\omega \times \omega$ which is onto J_α for those $\alpha < \Sigma$ (it is not uniform for all $\alpha \in \mathrm{On}$). Granting this we can define $\langle i, n\rangle \backsim \langle j, m\rangle$ iff $h(i, n) = h(j, m)$ and $\langle i, n\rangle E \langle j, m\rangle$ iff $h(i, n) \in h(j, m)$. Both $\backsim$ and E are (ordinary) recursive in T^2_α. Then $\langle [\langle i, n\rangle]_\backsim, E\rangle \simeq \langle J_\alpha, \in\rangle$. (ii) For i, n let $\pi : \omega \times \omega \longleftrightarrow \omega$ be a recursive pairing function, and then set $f(\pi(i, n)) = h(i, n)$. QED

Lemma 1.11. *There is an ittm programme $P_e = TM$ which does not converge, but continuously produces alternately codes x_α for levels J_α and their Σ_2-theories T^2_α for $\alpha < \Sigma$. At stage Σ as $T^2_\Sigma = T^2_\zeta$ TM loops back and reproduces the code x_ζ and continues this process thereafter repeating codes and theories for $\alpha \in [\zeta, \Sigma)$.*

Proof: We describe the effective procedure to be formalised. The input to TM is presumed to be zero. We let $\langle \varphi_n\rangle_{n\in\omega}$ be an effective enumeration of the sentences of $\mathcal{L}_\in$. We use the J-hierarchy to avail ourselves of uniform Σ_2-Skolem functions. This is not terribly important, but using the L-hierarchy is a bit more awkward. Recall that if $\omega \cdot \alpha = \alpha$ then $J_\alpha = L_\alpha$. On the output tape a theory T is written with $\varphi_n \in T$ iff the n'th cell contains a 1. In the first $\omega^2 + \omega \cdot 2$ stages TM writes the code of $J_1 = L_\omega = \mathrm{HF}$ and its Δ_0-diagram to two reserved tapes, and its Σ_2-theory to the output tape (ot). (It takes less than this, but it keeps the induction bookkeeping straight.) We assume inductively that at time $\omega^2 \cdot \alpha + \omega \cdot 2$ the ot contains the theory T^2_α of $\langle J_\alpha, \in\rangle$ and the reserved tapes again the Δ_0-diagram of J_α, d_α, and a code for J_α. With the theory T^2_α of J_α TM can construct a code $x_{\alpha+1}$ for $J_{\alpha+1}$ in ω^2 additional steps together with its Δ_0-diagram $d_{\alpha+1}$ (see Exercise). We are are now at stage $\omega^2 \cdot \alpha + \omega \cdot 2 + \omega^2$. In an additional $\omega \cdot 2$ steps $T^2_{\alpha+1}$ is calculated from $d_{\alpha+1}$ and written to OT as follows. This will take us to stage $\omega^2 \cdot (\alpha + 1) + \omega \cdot 2$. (This all takes some routine work to make clear, but essentially once we have T^0_β then T^{n+1}_β is r.e. in T^n_β, and so in particular $T^{n+1}_\beta \leq_T (T^n_\beta)'$. Each jump can be written out by an ittm in ω-steps (in fact the double jump can be so written, but we can ignore that), thus requiring $\omega \cdot 2$ steps to write out the two jumps and thus obtain the complete theory T^2_β.)

Of course we do this writing simply by changing the cells one by one according to what has appeared or disappeared passing from T^2_α to $T^2_{\alpha+1}$. If φ_n is in both theories, then the 1 in the n'th cell is not changed to a 0 and then back again to a 1. By this method of writing, at a limit stage $\omega^2\cdot\lambda$ for $\mathrm{Lim}(\lambda)$, $\widehat{T^2_\lambda}$ is on the ot, and thus the true T^2_λ is r.e. in the ot, by the first lemma. Hence in ω further steps it can then write the correct T^2_λ to the ot, thus by stage $\omega^2\cdot\lambda+\omega$, and then by the last lemma the code x_λ for J_λ on the scratch tape by stage $\omega^2\cdot\lambda+\omega+\omega$. A code for $J_{\lambda+1}$ and the diagram $d_{\lambda+1}$ is written by stage $\omega^2\cdot(\lambda+1)$, and $T^2_{\lambda+1}$ by $\omega^2\cdot(\lambda+1)+\omega\cdot 2$. QED

Exercise 1.2. Use the definition of the J_α-hierarchy using rudimentary closure, and fill in the details that a code for $J_{\alpha+1}$ can be written in ω^2 steps from a code for J_α simultaneously with its Δ_0-diagram. Use the fact that $J_{\alpha+1}=\mathrm{rud}(J_\alpha)$, and that there are 16 rudimentary basis functions under whose closure we can generate $\mathrm{rud}(J_\alpha)$. Having the Σ_2-theory, means we have the graph of h^2_α and so in effect, a partial onto map $f\colon\omega\longrightarrow J_\alpha$ with "$f(n)\in f(m)$" *etc.* recursive in T^2_α. [There are quite a lot of details here: assume we have a copy of x_α written on the Evens: $\langle 2n, 2m\rangle\in E_{x_\alpha}\longleftrightarrow f(n)\in f(m)$. Use the Odds as a space to build up the rudimentary closure of $J_\alpha\cup\{J_\alpha\}$, by ω many passes through the basis functions applied to whatever has been created so far. This creates the domain of $J_{\alpha+1}$. On another reserved tape simultaneously write the Δ_0-diagram of the sets being created. To do this use the fact that for any Δ_0 $\varphi(v_0,\ldots v_n)$ there is a rudimentary function F_φ (thus a combination of basis functions) so that $\varphi[x_0,\ldots x_n]$ iff $F_\varphi(x_0,\ldots x_n)\neq 0$.]

Exercise 1.3. (Lengthy but routine.) Adapt the above to show that there is an L-Theory Machine which works on the L rather than the J-hierarchy. [The problem with the L-hierarchy is that it is not closed under some basic operations such as Kuratowski-pairing as Boolos, Jensen *et al.* noticed and Jensen removed with the J-hierarchy. Boolos showed [1.1], that there are Σ_n-skolem functions for any L_α definable independent of $\alpha\geq\omega$, using uniformly definable wellorders of each L_α, as well as a sentence σ, so that for any transitive set $(M,E)\vDash\sigma\Longleftrightarrow\exists\alpha((M,E)=(L_\alpha,\in))$. Lemma 1.10 holds verbatim for L_α when $\mathrm{Lim}(\alpha)$ (indeed in Jensen's terms, for $\mathrm{Lim}(\alpha)$ L_α is rudimentarily closed). By Boolos, there is some $2\leq m<\omega$ so that for successor α there is still a uniform process in ω steps for producing a code x_α for L_α but now recursive in T^m_α, for $\alpha<\Sigma$ (potentially $2<m$ as Boolos's skolem functions for Σ_2 formulae are Σ_m-definable, and we have only that there is a $\Sigma^{L_\alpha}_m f\colon\omega\longrightarrow L_\alpha$, onto). Show that Lemma 1.11 still holds for T^m_α theories. Now assume inductively that at time $\omega^2\cdot\alpha+\omega\cdot m$ the ot contains the theory T^m_α of $\langle L_\alpha,\in\rangle$ and the reserved tapes again the Δ_0-diagram of L_α, d_α, and a code for L_α. Next adapt the last Exercise to show that from a code for L_α and its Δ_0-diagram we can still write a code for $L_{\alpha+1}$ in ω^2 steps, here using a fixed enumeration of the formulae with one free variable $\varphi_k(v_0)$ say, we may in ω-steps enumerate the set φ_k defines over L_α. With k increasing we may after ω^2 steps construct a code as required, noting that we keep within the same number of ordinals steps. At limit stages we still have $\widehat{T^2_\lambda}$ and all is as before.]

Corollary 1.12. *For any e, any real that appears at some stage on the output tape of $P_e(0)$ is recursive in some T^2_α for an $\alpha<\Sigma$, and thus is in L_Σ. Conversely for any real y of L_Σ there is an index e with y appearing on the OT of $P_e(0)$ at some stage (which perforce must be $<\Sigma$).*

Proof: Any course of computation of a $P_e(0)$ is absolute to $L_\Sigma\,(=J_\Sigma)$, with any snapshot at time s_α being definable over L_α ($\langle s_\beta|\ \beta<\alpha\rangle$ is defined by a Σ_1-recursion over L_α). Consequently the cell contents at stage α are recorded by a certain recursive subset of the theory T^2_α. Conversely if $y\in L_\Sigma$ then y is (ordinary) Turing recursive in a code x_α for some $\alpha<\Sigma$: for some $f\in\omega$, $y=\{f\}^{x_\alpha}$. Given f it is easy to amend TM so that the results of $\{f\}^{x_\alpha}$ are instead continuously written to the OT for increasing α. QED

Similar reasoning yields:

Corollary 1.13. *For any e, if $P_e(0)\downarrow y$, then y is recursive in some T^2_α for an $\alpha<\zeta$, and thus is in L_ζ. Conversely for any real y of L_ζ there is an index e with y $P_e(0)\downarrow^\alpha y$ at some stage $\alpha<\zeta$.*

Exercise 1.4. If λ is admissible, show that $\widehat{T^2_\lambda}=T^2_\lambda$. Consequently at time λ the theory T^2_λ for $J_\alpha(=L_\alpha)$ is on the ot.

Definition 1.14. *Let $\lambda<\zeta$ be least so that $L_\lambda\prec_{\Sigma_1}L_\zeta$.*

1.1. Boolos "*On the Semantics of the Constructible Levels*", Zeitschrift fuer Mathematische Logik, 1970, vol.16,139-148.

Lemma 1.15. *For any $P_e(0)$, if this computation formally halts at time τ, then $\tau < \lambda$. Conversely there are unbounded in λ ordinals τ for which there is such an e with $P_e(0)$ halting at time τ.*

Proof: By considering the Σ_2-recursion in L_ζ that yields the snapshots s_α, that s_τ is a halting snapshot of $P_e(0)$ is then a Σ_1 sentence in $T^1_{\tau+1}$ which is true in L_ζ but first true at $L_{\tau+1}$. By definition of λ then $\tau < \lambda$. Conversely there are unbounded in λ levels L_τ of the L-hierarchy where a new Σ_1 sentence σ_τ becomes true. (There could not be a bound $\lambda' < \lambda$ for such τ as then we should have $L_{\lambda'} \prec_{\Sigma_1} L_\lambda$.) However then we could run a program that itself runs TM and halts when it finds that $\sigma_\tau \in T^2_\tau$. This it can do by stage $\omega^2 \times (\tau + \omega) < \lambda$ for example, as λ is p.r. closed (in fact admissible). QED

Exercise 1.5. Amend the last program so that if $\tau < \lambda$ then there is a program $P_e(0)$ which halts with a code for a wellordering of type τ on its OT. Such a τ is then "ittm-writable".

As a corollary to the above we have:

Theorem 1.16. *(The "λ-ζ-Σ" Theorem [15]) Let ζ be the least Σ_2-extendible ordinal, with L_Σ the unique Σ_2 end-extension of L_ζ. Let λ be as in Def. 1.14. Then we have that:*

(i) (λ,ζ,Σ) is the lexicographic least triple with $L_\lambda \prec_{\Sigma_1} L_\zeta \prec_{\Sigma_2} L_\Sigma$;

(ii) $\lambda = \sup\{\tau | \tau$ is the halting time of some $P_e(0)\}$
$= \sup\{\tau | \tau = \|y\|$ is the length of some ordinal code y output by a halting program $P_e(0)\}$.

(iii) $\zeta = \sup\{\tau | \tau$ is the convergence time of some $P_e(0)\}$
$= \sup\{\tau | \tau = \|y\|$ where $\exists e\ P_e(0)\!\downarrow y\}$.

(iv) $\Sigma = \sup\{\tau | \tau = \|y\|$ where $\exists e$ with y appearing on the OT of $P_e(0)$ at some stage $\tau < \Sigma\}$.

• In the literature an ordinal is "clockable" if it is the halting time of some $P_e(0)$. We thus have in (ii), using Ex.1, that all "clockables are writables". The ordinals τ (or reals y) in (iv) are called "accidental", and those in (iii) "eventually writable". See, *e.g.* [16].

1.3 Infinite Time Jump operator

Definition 1.17. *(The infinite time jump* iJ*)*
(i) We write $\{e\}(\boldsymbol{m}, \boldsymbol{x})\!\downarrow$ if the e'th ittm-computable function with input $\boldsymbol{m}$, $\boldsymbol{x}$ has a fixed output $c \in 2^{\mathbb{N}}$, in which case we write $\{e\}(\boldsymbol{m}, \boldsymbol{x}) = c$.
(ii) We then define iJ *by:*

$$\text{iJ}(e, \boldsymbol{m}, \boldsymbol{x}) = \begin{cases} 1 & \text{if } \{e\}(\boldsymbol{m}, \boldsymbol{x})\!\downarrow; \\ 0 & \text{otherwise (for which we write } \{e\}(\boldsymbol{m}, \boldsymbol{x})\!\uparrow). \end{cases}$$

$$\text{iJ}(y) = y \text{ if } y \text{ is not of the form } \langle e, \boldsymbol{m}, \boldsymbol{x}\rangle.$$

The functional iJ then is the counterpart of the standard tm operator oJ.

Definition 1.18. *For x a real, the complete (ordinary)* ittm-semirecursive-in-x set*, denoted by $\tilde{x}$ is the set of integers $\{e | \{e\}(e, x)\!\downarrow\}$.*

Exercise 1.6. It is a consequence of the (relativized to x) λ-ζ-Σ-Theorem above that $\tilde{x}$ is recursively isomorphic to the complete Σ_2-Theory of $L_{\zeta^x}[x]$. Show this.

2 Generalised type-2 ittm-recursion

2.1 Generalised ittm-recursion in a type 2 functional I

In the Kleenean recursion in type-2 functionals, in [9],[10] (building up an equivalent approach to [8] and [11]) a successful computation (meaning one with output) could be effected by imagining tm's placed at nodes on a wellfounded tree, with computations proceeding at nodes that make computation calls to a lower node, seeking the value of some $x(k)$ say. The computation time at each node, regarding each call to a lower node as being just one step in the computation of the calling node, is then finite. (For otherwise the computation at the node is never completed and the whole overall computation will fail.) An overall computation may also fail by instituting a series of calls to subcomputations that form an infinite descending path in the tree. In such cases the machines on the path all hang after finitely many steps, all waiting for data to be passed up from the immediate subcomputation it has called.

In the ittm case we may again conceive of an overall or master ittm computation taking place at the top level; such a computation may take infinitely many steps in time, and will be considered as successful if the output tape is fixed from some point in time onwards. The master computation may make queries of a type-2 functional I in which the computation is considered ittm-recursive. It may call subcomputations of exactly the same type: ittm's with the capability to make oracle queries of I.

We give a more detailed description of this as a representation in terms of underlying ittm's. $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ will represent the e'th program in the usual format, say transition tables, but designed with appeal to oracle calls possible. We are thus considering computations of a partial function $\{e\}^{\mathsf{I}}\colon {}^{k}\omega \times {}^{l}({}^{\omega}2) \to \omega$. Such a computation has potentially computation time, or stages, unbounded in the ordinals.

The computation of $P_e^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ proceeds in the usual ittm-fashion, working as a tm at successor ordinals and taking $\liminf$'s of cell values *etc.* at limit ordinals. At a time α an oracle query may be initiated. We may conventionally fix that the real number subject to query is that infinite string on the even numbered cells of the scratch type. If this string is $(f, m, y_0, y_1 \ldots,)$ then setting $y = y_0, y_1, \ldots$, the *query* or *oracle call* which we shall denote $Q^{\mathsf{I}}(f, m, y)$ is the question: ?*What is* $\mathsf{I}(z)$ *where* $P_f^{\mathsf{I}}(m, y)\!\downarrow\! z$? and at stage $\alpha + 1$ receives the value $\mathsf{I}(z)$. If it is not the case that $P_f^{\mathsf{I}}(m, y)\!\downarrow z$ for any z, *i.e.*, it fails to have a fixed output, then there is no z to which I can be applied, and the overall computation fails. (We could try to stay closer to the Kleenean setting, where a tree branches infinitely often downwards, to potentially compute some $z \in {}^{\omega}\omega$, *via* $z(0), z(1), \ldots$ in turn, and then can ask for $\mathsf{I}(z)$. There, if any one of the single computations $z(k)$ failed, then the query to I did not take place, and the overall computation failed. But one thing we have with ittm computation is plenty of time, so we can, and do, amalgamate the individual computations $z(k)$ as simply one computation of all of z.)

We can determine its effect as follows *via* an inductive operator I. Just as the Kleene equational calculus can be seen to build up in an inductive fashion a set of indices and equational strings $\Omega[\mathsf{I}]$ for successful computations recursive in I (see Hinman [6], pp. 259-261), so we can define the fixed point of a monotone operator $I = I^{\mathsf{I}}$ on $(\omega \times \omega^{<\omega} \times (\omega^{\omega})^{<\omega}) \times \omega^{\omega}$ which will give us the successful ittm-computations recursive in I. (We blur distinctions between Cantor and Baire space. Note that we have defined the outputs here as reals in Baire space, rather than just integers in ω which the notion of a Type-2 functional would seem to require. However with ittm's, just as in the comment above, to compute a $z \in \omega^{\omega}$ is just to compute the sequence $z(0), z(1), \ldots$ which we can do here, and may consider the characteristic function of the graph of z as an element of Cantor space. So by doing this we simply acknowledge that fact of life for ittm's.

Definition 2.1. *We set* $I(X) =:$

$$\{\langle\langle e, \boldsymbol{m}, \boldsymbol{x}\rangle, z\rangle \mid P_e^X(\boldsymbol{m}, \boldsymbol{x})\!\downarrow\! z \text{ is an ittm-computation making only oracle calls } Q^X(e', \boldsymbol{m'}, \boldsymbol{x'}) \text{ and receiving back } \mathsf{I}(z') \text{ where } X(\langle e', \boldsymbol{m'}, \boldsymbol{x'}\rangle) = z'\}.$$

As this is monotone, we may let

$I^0 = \varnothing$; $I^{<\alpha} = \bigcup_{\beta<\alpha} I^{\beta}$ & $I^{\alpha} = I(I^{<\alpha})$ *in the usual way, and reach a least fixed point* I^{∞}.

Definition 2.2. *The* rank *of a defined computation, $\rho^{\mathsf{I}}(\langle\langle e, \boldsymbol{m}, \boldsymbol{x}\rangle, z\rangle)$ is the least α, if it exists, such that $\langle\langle e, \boldsymbol{m}, \boldsymbol{x}\rangle, z\rangle \in I^{\alpha}$. We often abbreviate this as $\rho^{\mathsf{I}}(e, \boldsymbol{m}, \boldsymbol{x})$ with the z understood but unspecified.*

Then:

Theorem 2.3. (The $\{e\}$'th function partial generalised-ittm-recursive in I) *Using I^{∞}:*

$\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ *is* defined, *or* convergent, *with output z iff $I^{\infty}(\langle e, \boldsymbol{m}, \boldsymbol{x}\rangle) = z$.*

In which case we set $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x}) = z$ or write $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})\downarrow z$. Otherwise it is undefined. *$\{e\}^{\mathsf{I}}$ is* generalised-ittm-recursive in I *if it is partial generalised-ittm-recursive in I and total.*

Definition 2.4. *For functionals I, J we say $\mathsf{I} \leq \mathsf{J}$ ("I is* (ittm-generalised) partial recursive in J*") if there is $e \in \mathbb{N}$ so that $\mathsf{I} = \{e\}^{\mathsf{J}}$. We write $\mathsf{I} \equiv \mathsf{J}$ if both $\mathsf{I} \leq \mathsf{J}$ and $\mathsf{J} \leq \mathsf{I}$ hold.*

The functional I is recursive in J *if it is partial recursive in J and total. A relation R is* recursive in J *if the characteristic function K_{R} is recursive in J.*

(ii) A relation R is semi-recursive in J *if it is the domain of a function partial recursive in J.*

Exercise 2.1. Show that R is recursive in I iff both R and its complement is semi-recursive in I.

Exercise 2.2. Show that the class of relations semi-recursive in a functional I is closed under both universal and existential number quantification $\forall^{\omega}$ and $\exists^{\omega}$, and in particular under $\cap$ and $\cup$.

Definition 2.5. *The (top-level) length of a computation $P_e^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ in a type-2 oracle I is the least $\sigma_0 = \sigma_0^{\langle \mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x}\rangle}$ (when defined) so that the snapshot at time σ_0 of $P_e^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ is the repeat of some earlier snapshot $\zeta_0 = \zeta_0^{\langle \mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x}\rangle} < \sigma_0$, and so that the snapshot at σ_0 recurs unboundedly in* On.

Again, by an easy Loewenheim-Skolem argument, the ordinal σ_0 is countable. Thus the snapshots of the cell distributions in $(\zeta_0, \sigma_0]$ form the final loop which infinitely repeats thereafter. Actually this top-level length is of less interest than the *overall length* of the computation - to be defined below. Both of these will be undefined if the computation tree describing $P_e^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ is illfounded. We give here a more detailed description of these trees.

Continuing the discussion above, the $\{f\}^{\mathsf{I}}$'th function on input m, y say, has the opportunity to make oracle calls, and we shall thus have a *tree* representation of calls made. We wish to represent the overall order of how such calls are made, and indeed the ordinal times of the various parts of the computation as it proceeds.

Computation trees $\mathfrak{T} = \mathfrak{T}^{\mathsf{I}}(e, \boldsymbol{m}, \boldsymbol{x})$

Overall we have a 'linear' mode of evaluation of the *computation tree* - also called a *tree of subcomputations*. In particular we should like to keep track of an *overall length of computation*. This overall length will be the length not of the top node only, (which we may think of as the 'master computation', and receives its replies to oracle queries immediately in one step only) but as of the whole computation when the lengths of the computations at lower nodes of the tree, which we regard as actually performing the sub-computations of the form $P_f^{\mathsf{I}}(y)\downarrow z$ in order to obtain $\mathsf{I}(z)$, are then also taken into consideration. It will rapidly be seen that the structure of the tree $\mathfrak{T} = \mathfrak{T}^{\mathsf{I}}(e, \boldsymbol{m}, \boldsymbol{x})$ of a convergent computation $P_e^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})\downarrow z$ reflects how subcomputations arrive into the fixed point of the induction Def. 2.1, and thus the rank of this wellfounded tree will be the ordinal $\rho^{\mathsf{I}}(\langle\langle e, \boldsymbol{m}, \boldsymbol{x}, z\rangle\rangle)$. Thus although the computation is most easily represented by a tree, we may think of the computation as a linear process as we visit each node of the tree in turn.

We therefore make the following conventions. During the calculation of $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ the initial calculation takes place at the topmost node ν_0 which we declare to be *at Level* 0, in our computation tree $\mathfrak{T} = \mathfrak{T}^{\mathsf{I}}(\boldsymbol{e}, \boldsymbol{m}, \boldsymbol{x})$. (We set $e_0 = e$, $n_0 = \boldsymbol{m}$, $y_0 = \boldsymbol{x}$ and pretend that this computation and all the oracle calls below are only for single number and real variable, merely for ease of presentation.) Let us suppose the first instruction for an oracle query concerning $\{e_1\}^{\mathsf{I}}(n_1, y_1)$ is actioned at stage δ_0. The tree $\mathfrak{T}$ will then have a node ν_1 below ν_0, labelled with $\langle e_1,\ n_1,\ y_1\rangle$ and we declare the computation $\{e_1\}^{\mathsf{I}}(n_1,\ y_1)$ to be performed at this Level 1. Thus 'control' of the overall process is defined to be at the level of the node ν_1 at stage $\delta_0 + 1$. The 'time' for this sub-computation, starting thus at $\delta_0 + 1$ of course starts locally at its '$t = 0$' - although each stage is also thought of as one more step in the overall length of the computation above: namely of $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$. Suppose $\{e_1\}^{\mathsf{I}}(n_1, y_1)$ makes no further oracle calls and the least stage at which it exhibits looping behaviour, according to Def. 1.5 is σ_1. If there is a settled output of $\{e_1\}^{\mathsf{I}}(n_1, y_1)$, z say, then the correct value $\mathsf{I}(z)$ is then passed back up to Level 0 at the next stage, that is $\delta_0 + 1$ *but only in terms of the stages of computation at the top level*, and the master computation proceeds to its next step at this Level 0.

However we deem that $\delta_0 + 1 + \sigma_1 + 1$ steps have occurred so far towards the final *overall length*, or $H = H(e, \boldsymbol{m}, \boldsymbol{x})$ of the calculation, that is, of what will be $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ if it is successful. To be clear: at stages $\delta \in (\ \delta_0 + 1 + \sigma_1]$ the computation is at ν_1, whilst in the interval $[0, \delta_0]$ and at $\delta_0 + 1 + \sigma_1 + 1$ it is at ν_0.

However if $\{e_1\}^{\mathsf{I}}(n_1,\ y_1)$ itself has made an oracle query, let us suppose the first such was $?Q^{\mathsf{I}}(e_2, n_2, y_2)?$, then a new node ν_2 is placed below ν_1 labelled with $\langle e_2, n_2, y_2\rangle$ (the label is also part of $\mathfrak{T}$). If this piece of computation at ν_2 is successful, that it has settled output z' say, and if we suppose it made no oracle calls, and took σ_2 steps to exhibit looping behaviour, then the value $\mathsf{I}(z')$ is passed back up to ν_1; lastly the overall length of $\{e_2\}^{\mathsf{I}}(n_2,\ y_2)$ is σ_2 and then those σ_2 steps will have to be added to the overall length of calculation for $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$, being added as they are, to the top-level length of $\{e_1\}^{\mathsf{I}}(n_1, y_1)$.

If the computation $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ is defined then we shall have as its computation tree $\mathfrak{T} = \mathfrak{T}(e, \boldsymbol{m}, \boldsymbol{x})$ a finite path tree (with potentially infinite branching) and some countable rank. $\mathfrak{T}$ will be labelled with nodes $\{\nu_\iota\}_{\iota < \eta(\mathfrak{T})}$ that are visited by the computation in increasing order (with backtracking up the tree of the kind indicated). Thus ν_ι is first visited only after all ν_τ have been visited for $\tau < \iota$. (This is the sense in which the computation can be considered as linear after all.) The β'th oracle call to Level k will generate a node we picture as placed to the right of those so far at Level k (meaning to the right of those with lesser indices $\alpha < \beta$ at that level). When a subcomputation at a node successfully finishes, then control of the overall computation is envisaged as passing one level up to the node immediately above. As the computation progresses it traverses the tree in the order of the indices on the nodes just described. We could say that *'control'* of the process is *at a node* ν_ι (or is *at a level*) *at time* t in the overall length, if the current sub-computation is running at the node (or at a node at that level) at that time t. (See the next definition.)

The tree will thus have a linear leftmost branch, before any branching occurs. Further, for a well-founded tree we may define the *overall length function* $H = H(\mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x})$ as above, formally by recursion on the rank of nodes as the length of the computation.

Definition 2.6. *(The overall length function)* $H = H(\mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x})$ *is defined by recursion on the rank* $\rho^{\mathsf{I}}(e, \boldsymbol{m}, \boldsymbol{x})$. *Let* $\sigma_0 = \sigma_0^{\langle \mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x}\rangle}$ *and suppose that* $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ *makes sub-computation calls* $\{e_\iota\}^{\mathsf{I}}(n_\iota, y_\iota)$ *at times* $\tau_\iota < \sigma_0$ *for* $\iota < \theta \leq \sigma_0$. *Then*

$$H(\mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x}) =_{\mathrm{df}} \sum_{\iota<\theta} \left((\tau_\iota - \sup\{\tau_\xi |\ \xi < \iota\}) + 1 + H(\mathsf{I}, e_\iota, n_\iota, y_\iota)\right) + (\sigma_0 - \sup\{\tau_\iota |\ \iota < \theta\}).$$

Then H gives simply the total ordinal length of the whole computation together with its subcomputations as if laid out in a linear fashion.

Definition 2.7. *(i) The* level *of a node ν_ι is the length of the path in the tree from ν_0 to ν_ι.*
(ii) By Level n *we accordingly mean the set of nodes in the tree with level n.*
(iii) The node of the computation $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ at time $\alpha < H(\mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x})$, *denoted $\nu(\alpha) = \nu(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \alpha)$, is the node ν_ι at which the overall computation is being performed at time α, and has label $\langle e_{\nu(\alpha)}, \boldsymbol{m}_{\nu(\alpha)}, \boldsymbol{x}_{\nu(\alpha)}\rangle$. The* level $\Lambda(\alpha) = \Lambda(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \alpha)$ *is the level of $\nu(\alpha)$.*
(iv) The current snapshot at time $\alpha < H(\mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x})$ *is denoted $\langle I(\alpha), R(\alpha), \langle C_i^{\nu(\alpha)}(\alpha) | \, i < \omega\rangle\rangle$, and equals the snapshot $\langle I(\bar\alpha), R(\bar\alpha), \langle C_i(\bar\alpha) | \, i < \omega\rangle\rangle$ of the sub-computation $\{e_{\nu(\alpha)}\}(\boldsymbol{m}_{\nu(\alpha)}, \boldsymbol{x}_{\nu(\alpha)})$, where $\bar\alpha = \alpha - \pi_{\nu(\alpha)}$, and which was initiated at the overall time $\pi_{\nu(\alpha)}$.*
If $\mathsf{I} = \mathsf{E}$ (see Def. 2.11 below) it can be omitted.

Thus for a *defined* (or '*successful*') computation, at any time the level is a finite number ('depth' would have been an equally good choice of word). An *undefined,* or *failed*, or *unsuccessful*, *computation* is one in which a sub-computation call resulting in a calculation at some node fails to produce an output z (and so no value $\mathsf{I}(z)$ can be returned to the level above) either (a) because some sub-computation produced no convergent output or (b) $\mathfrak{T}(\mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x})$ is illfounded (with a rightmost path of order type then ω); or else (c) the topmost computation itself fails to have convergent output, *i.e.* to have a stable output tape. In (iv) the current snapshot is thus the snapshot of the machine that is running at time α (thus computing $\{e_{\nu(\alpha)}\}(\boldsymbol{m}_{\nu(\alpha)}, \boldsymbol{x}_{\nu(\alpha)})$). This machine started running when it's local time was $t = 0$ of course, but in the overall picture of things, it starts at time $\pi_{\nu(\alpha)}$, and $\bar\alpha$ simply gives how many steps it has run at overall time α.

The following lemma incorporates level and cell value facts from the description of the trees and how control passes from level to level just given. The point to notice, *e.g.* in *(i)*, is if a computation at a node $\nu(\alpha)$ locally reaches its first repeating point σ_α say, then control is immediately passed back to one level above; thus ν and Λ is decreased. So the liminf in question in *(i)* for Λ is over levels of computation at the current level or below in the tree, and thus cannot contribute unboundedly in λ smaller integers to the liminf. Similar considerations justify that $\nu(\lambda) = \mathrm{liminf}_{\alpha\to\lambda}\nu(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \alpha)$: any subcomputation calls are to nodes with higher nodal index: to $\nu(\beta)$ greater than what will be $\nu(\lambda)$ on a tail of β below λ.

Lemma 2.8. *Let* $\mathrm{Lim}(\lambda)$. *The computation $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$, if not divergent by stage λ, satisfies:*
(i) $\nu(\lambda) = \nu(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \lambda) = \mathrm{liminf}_{\alpha\to\lambda}\nu(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \alpha)$;
and so: $\Lambda(\lambda) = \Lambda(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \lambda) = \mathrm{liminf}_{\alpha\to\lambda}\Lambda(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \alpha)$
(ii) If $\nu = \nu(\mathsf{I}, e, (\boldsymbol{m}, \boldsymbol{x}), \lambda)$, and if $\{e_\nu\}^{\mathsf{I}}(\boldsymbol{m}_\nu, \boldsymbol{x}_\nu)$ is the subcomputation at Level $k = \Lambda(\lambda)$, currently being run at stage λ, then if $\langle C_i^\nu | \, i < \omega\rangle$ are the cell values of this subcomputation, then $C_i^\nu(\lambda) = \mathrm{liminf}_{\alpha\to\lambda, \nu(\alpha)=\nu} C_i^{\nu(\alpha)}(\alpha)$.

Exercise 2.3. Find an index e_1 so that $\{e_1\}^{\mathsf{I}}(0)\!\downarrow$ in $<\omega$ steps (at the top level) for any I, but $H(e_1, 0, 0) \geq \zeta$.

Exercise 2.4. Find an index e_2 so that $\{e_2\}^{\mathsf{I}}(0)\!\uparrow$ (for any I).

Usual methods prove an S_m^n-Theorem and more particularly:

Theorem 2.9. (The generalised ITTM-Recursion Theorem) *If $F(e, \boldsymbol{m}, \boldsymbol{x})$ is ittm-recursive in I, there is $e_0 \in \omega$ so that*

$$\{e_0\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x}) = F(e_0, \boldsymbol{m}, \boldsymbol{x}).$$

Lemma 2.10. *The computation $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$ is absolute to $L[\mathsf{I}, \boldsymbol{x}]$. In the above notation, there is a function $S(\alpha) = S(\mathsf{I}, \boldsymbol{m}, \boldsymbol{x}, \alpha)$ for $\alpha \in \mathrm{On}$, (the 'snapshot function') so that*
(i) $S(\beta) = \langle \nu(\beta), \Lambda(\beta), \langle I(\beta), R(\beta), \langle C_i^{\nu(\beta)} | \, i < \omega\rangle\rangle\rangle$; (ii) $\langle S(\beta) | \, \beta < \alpha\rangle \in \Delta_1^{J_\alpha[\mathsf{I}, \boldsymbol{x}]}$;

(iii) $S(\alpha) \in \Sigma_2^{J_\alpha[\mathsf{I},\boldsymbol{x}]}$.

2.2 The functional $^2\mathsf{E}$

This is the simple functional of existential quantification. Recall that we are representing elements of Baire space, so $x \in {}^\omega\omega$, in ${}^\omega 2$ on the tape as the infinite sequence of 1's interspersed with a string of 0's of length $x(n)+1$.

Definition 2.11. *(The functional* $^2\mathsf{E}$*) (i) We define* $\mathsf{E} = {}^2\mathsf{E}\colon {}^\omega\omega \longrightarrow {}^\omega\omega$ *by:*

$$\mathsf{E}(x) = \begin{cases} 1 & \text{if } x \in {}^\omega\omega \wedge \exists n\, x(n) \neq 0 \\ 0 & \text{if } x \in {}^\omega\omega \wedge \forall n\, x(n) = 0 \end{cases}$$

(ii) For $R \subseteq \omega$ *we set* $\mathsf{E}(R) =_{\mathrm{df}} \mathsf{E}(K_R)$.

Recursions in $^2\mathsf{E}$ are already quite powerful: Kleene showed that for Kleene recursion oJ and $^2\mathsf{E}$ are mutually Kleene recursive in each other. The functional E was important for much of the development of this recursion, and a type-2 functional I for Kleene was *normal* if $^2\mathsf{E}$ was Kleene-recursive in I. Many of the theorems of theory were only valid for normal functionals. For Kleene recursion the functionals oJ and $^2\mathsf{E}$ being equivalent, these were the simplest useful functionals. However here in the ittm setting normality is trivial, but again, with iJ and $^2\mathsf{E}$ being the simplest useful functionals. We shall see that for ittm-recursion $^2\mathsf{E}$ and iJ are likewise mutually recursive. (Note that $^2\mathsf{E} \leq \mathsf{iJ}$ is trivial. Check!)

For performing a computation together with all its subcomputations as a tree, and seeing how the length of computation relates to extendibility in the L hierarchy, even if the function I is quite simple, and constructibly definable, this may have to be done in suitably large admissible sets. However note that any computation $\{e\}^{\mathsf{E}}(\boldsymbol{m},\boldsymbol{x})$ is absolute to $L[\boldsymbol{x}]$.

We have relativized in Definition 1.3 to reals x in the obvious way, the concept of x-(Σ_2)-extendible pairs (ξ,σ). Note that for such a pair, since E is Δ_1-definable over L_ξ, so (ξ,σ) is also an x-E-(Σ_2)-extendible pair in an obvious sense. We use this without further mention.

Lemma 2.12. *If* (ξ,σ) *is an* $\boldsymbol{x}$*-extendible pair, then*
(i) $\nu(e,(\boldsymbol{m},\boldsymbol{x}),\xi) = \nu(e,(\boldsymbol{m},\boldsymbol{x}),\sigma)$ *and so* $\Lambda(e,(\boldsymbol{m},\boldsymbol{x}),\xi) = \Lambda(e,(\boldsymbol{m},\boldsymbol{x}),\sigma)$*;*
(ii) If $\nu(e,(\boldsymbol{m},\boldsymbol{x}),\sigma) = \nu$ *then in the notation above* $C_i^\nu(\xi) = C_i^\nu(\sigma)$ *for* $i < \omega$*;*
(iii) If in (ii) $\nu = \nu_0$*, then* $\{e\}^{\mathsf{E}}(\boldsymbol{m},\boldsymbol{x})$ *has entered final looping behaviour by stage* ξ*.*

Proof: These all follow from the Σ_2 Liminf nature listed in Lemma 2.8 of the properties of Def. 2.7. QED

Conversely:

Lemma 2.13. *The computation* $\{e\}^{\mathsf{E}}(\boldsymbol{m},\boldsymbol{x})$ *is absolute to* $L[\boldsymbol{x}]$*. If* $\{e\}^{\mathsf{E}}(\boldsymbol{m},\boldsymbol{x})$ *is convergent, then there is* (ξ,σ) *an* $\boldsymbol{x}$*-extendible pair with* $\Lambda(e,(\boldsymbol{m},\boldsymbol{x}),\xi) = \Lambda(e,(\boldsymbol{m},\boldsymbol{x}),\sigma) = 0$.

Definition 2.14. *A type-2 functional* I *is called* suitable *if the* $\mathrm{ran}(\mathsf{I} \restriction \omega)$ *is not a singleton, where we represent* $k \in \omega$ *as the infinite sequence of* $k+1$ *1's followed thereafter by* 0*'s.*

Here, ${}^{<\omega}\omega$ should be interpreted as those infinite strings from Cantor space that are zero from some point onwards. Thus by abuse of notation $\mathbb{N}$ "$\subseteq$" ${}^{<\omega}\omega$. Then iJ is suitable. It is easy to see that for any K there is a suitable $\mathsf{K}' \equiv \mathsf{K}$.

Lemma 2.15. *If* J *is suitable, there is* $\bar{e}$ *so that* $\{\bar{e}\}^{\mathsf{J}}(e, m, y) = 1$ *if* $\{e\}^{\mathsf{J}}(m, y)\downarrow$, *and* $= 0$ *if* $\{e\}^{\mathsf{J}}(m, y)\uparrow$ *(to emphasise, this* $\uparrow$ *does not mean that* $\{e\}^{\mathsf{J}}(m, y)$ *fails but rather has divergent OT); and is undefined if* $\{e\}^{\mathsf{J}}(m, y)$ *fails.*

Proof: We assume without loss of generality that $\mathsf{J}(\overline{0}) = 0$ and $\mathsf{J}(\overline{1}) = 1$. (It will be apparent what to do if we need to appeal to other values under suitability of J.) We define the following procedure P which will be realised as a programme $P^{\mathsf{J}}_{\bar{e}}$. The process P does the following: (A) it looks for a snapshot of the top level of the computation of $P^{\mathsf{J}}_e(m, y)$ that is a *repeating snapshot*. (B) When it finds one, it will have checked that it initiates a proper final loop in the computation of $P^{\mathsf{J}}_e(m, y)$. (C) A snapshot that passes the check at (B) can also at the same time be inspected to see if the OT that it encodes has a convergent value or not. In more detail:

1) P first runs a copy of the program $P^{\mathsf{J}}_e(m,\ y)$ on a scratch tape, running the programme instructions coded in e.

For each top level time α of the simulated run of this $P^{\mathsf{J}}_e(m,\ y)$, snapshots s_α of this top level tape *etc.* at times α are written by P to a reserved piece of tape R (each snapshot overwriting the contents of R as this is repeatedly done).

2') Then P makes the query $?Q^{\mathsf{J}}(t_0, (e, m, y, s_\alpha))?$ where the action of $T_0 =_{\mathrm{df}} P^{\mathsf{J}}_{t_0}(e, m, y, s_\alpha)$ does the following:

(i) T_0 first sets the first and second cells of OT(T_0) (the output tape of T_0) to 0;

(ii) T_0 itself runs the code of $\{e\}^{\mathsf{J}}(m, y)$ on a scratch tape, as in 1) but instead starting from the stage α onwards, using for this as input the snapshot data s_α from R, all the while inspecting the later snapshots s_β for $\beta \geq \alpha$ that it generates.

(iii) If the snapshot s_α recurs, with the proviso that no cell of s_α which has a 1, switches $1 \longrightarrow 0 \longrightarrow 1$, T_0 sets the first cell of $\mathrm{OT}(T_0)$ to 1.

(iv) If the OT of this simulation of $\{e\}^{\mathsf{J}}(m, y)$ changes then on the first such occasion the second cell of OT(T_0) is set to 1.

Otherwise it does nothing further to those first two cells, but continues the run.

Thus in every case we see that $P^{\mathsf{J}}_{t_0}(e,\ m,\ y,\ s_\alpha)$ is convergent, and then consulting J as an oracle concerning $\mathrm{OT}(T_0)$, the oracle returns a value back (by our assumptions on J) to P and this query is finished. By our assumptions on J, $\mathsf{J}(1{\frown}0) = 0$ indicates that $P^{\mathsf{J}}_e(m, y)\downarrow$ and $\mathsf{J}(1{\frown}1) = 1$ indicates that $P^{\mathsf{J}}_e(m, y)\uparrow$. Then the appropriate 1 respectively 0 can be written to the OT of P. On the other hand if $\mathsf{J}(0{\frown}i) = 2$ occurs then the control passes back to the start of 2') with P making the query $?Q^{\mathsf{J}}(t_0, (e, m, y, s_{\alpha+1}))?$ on the next snapshot $s_{\alpha+1}$. (Note that at limit stages μ of this process the liminf process naturally records the correct snaphot s_μ in R.)

Eventually P will reach a snapshot s_ξ that will indeed be the start of a final loop and J will be returning the correct $0/1$ value. QED

Exercise 2.5. If J is suitable, show there is $\bar{e}$ so that $\{\bar{e}\}^{\mathsf{J}}(e, m, y) = 1{\frown}z$ if $\{e\}^{\mathsf{J}}(m, y)\downarrow z$, and $= 0$ if $\{e\}^{\mathsf{J}}(m, y)\uparrow$.

Exercise 2.6. Show that there is a p.r. function f with

$$\{e_1\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x}, \lambda n.\{e_2\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x})) \simeq k \longleftrightarrow \{f(e_1, e_2)\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x}) \simeq k.$$

Deduce that the class of functionals partial recursive in I is closed under substitutions.

Exercise 2.7. For any I and any F and y, if x is recursive in I (and y), and F is partial recursive in I and x, then F is partial recursive in I (and y). $\{e\}^{\mathsf{I},x}$[XXX? To complete]

Exercise 2.8. Let J be suitable. Show that a query instruction $?Q^{\mathsf{J}}(t,\ n,\ y))?$ in a computation $P_e^{\mathsf{J}}(\boldsymbol{m},\ \boldsymbol{x})$, can be replaced with some finite set of instructions which initiates a recursive sequence of queries $?Q^{\mathsf{J}}(t_i,\ n,\ y))?$ which effects the writing to a recursive slice of the scratch tape R, of the sequence of digits $z(i)$ where, if it exists, $\{t\}^{\mathsf{J}}(n, y)\downarrow z$. (Thus instead of the query returning just the single integer $\mathsf{J}(z)$ we can think of the amended program as returning z itself to (a recursive slice of) the scratch tape. This 'subroutine' is independent of e, but with t_i prim. recursively dependent on t and i only.)

Exercise 2.9. Let J be suitable. Show that the behaviour of computations in J can be modified in the following way. Show that there is a p.r. function h so that in any computation $P_e^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x})$ any query call $?Q^{\mathsf{J}}(t, n, y))?$ that occurs throughout the computation tree, is replaced in $P_{h(e)}^{\mathsf{J}}(\boldsymbol{m},\ \boldsymbol{x})$ by one which returns $z \simeq P_t^{\mathsf{J}}(n,\ y)$ itself to some recursive slice of the scratch tape of the calling computation, before returning the integer $\mathsf{J}(z)$. [Hint: First show that there is a p.r. function g so that in a computation $P_e^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x})$ any query call $?Q^{\mathsf{J}}(t, n, y))?$ at the top level is replaced in $P_{g(e,i)}^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x})$ by one which instead returns the i'th value $z(i)$. Fixing a recursive slice R of the scratch tape, then show that these can be amalgamated to produce a piece of code that recursively calls for each value $z(i)$ is turn. This then writes z to R. Call this program $P_{\bar{h}(e)}$. Now, somewhat trivially do one more modification: write z as input to the identity programme $P_\iota^{\mathsf{J}}(z) = \mathrm{id}(z)$ and then make the call $?Q^{\mathsf{J}}(\iota,\ z)?$ This returns $\mathsf{J}(z)$ after all which can be written to its suitable piece of tape.

This yields a p.r. function h_0 which substitutes this code for the top level queries occurring in $P_e^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x})$, yielding a computation $P_{h_0(e)}^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x})$. Lastly use the recursion theorem to show that there is a p.r. function h so that $P_{h(e)}^{\mathsf{J}}(\boldsymbol{m},\ \boldsymbol{x})$ is the result of applying h_0 throughout the computation tree for $P_e^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x})$, to the indices of all sub-computation calls. By design we shall have that $P_e^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x}) \simeq P_{h(e)}^{\mathsf{J}}(\boldsymbol{m}, \boldsymbol{x})$, although the latter computation has done this extraneous work.]

Exercise 2.10. Let J be suitable. Show that there is a p.r. function p so that whenever $\{e\}^{\mathsf{J}}(m,\ y)$ has a wellfounded computation tree, then $\{p(e)\}^{\mathsf{J}}(m,\ y)\downarrow s$ where s is the finally looping snapshot of $\{e\}^{\mathsf{J}}(m,\ y)$.

Exercise 2.11.

Exercise 2.12.

Exercise 2.13. Show that there is a p.r. function f so that for suitable J , if $\{e\}^{\mathsf{J}}(m,\ y)$ is a computation in which a query $?Q^{\mathsf{J}}(e_1,\ m_1,\ x_1)?$ occurs, then $\{f(e)\}^{\mathsf{J}}(m,\ y)$ is a computation in which the query is replaced by an infinite sequence of queries with the result that if $\{e_1\}^{\mathsf{J}}(m_1, x_1)\downarrow z$ then the sequence has the effect of writing z to a segment of scratch tape of the calling the computation $\{e\}^{\mathsf{J}}(m,\ y)$. [Hint: Programs are finite so this cannot literally be an infinite list of queries, but instead is a subroutine with a recursive list of indices writing each digit of z in turn.]

Notice in the above, that the only dependence of $\bar{e}$ is on the way that J indicates its suitability. If J takes differing values on three other finite sequences this only alters what we require of the processes T, T', T'' to write on their OT's in the specification above.

Theorem 2.16. *If* $\mathsf{I} \leq \mathsf{J} \leq \mathsf{K}$ *and* K *is suitable, then* $\mathsf{I} \leq \mathsf{K}$. *More generally there is a p.r. function* f, *so that* $\{e\}^{\mathsf{J}} = \{f(e)\}^{\mathsf{K}}$.

Proof: We again assume without loss of generality that $\mathsf{K}(0) = 0$ and $\mathsf{K}(1) = 1$. (It will again be apparent what to do if we need to appeal to other values under suitability of K.) We are given that there are $g_1, g_2 \in \omega$ so that $\mathsf{I} = \{g_1\}^{\mathsf{J}}$ and $\mathsf{J} = \{g_2\}^{\mathsf{K}}$. We show there is a method for finding an index g_3 for a recursion $\mathsf{I} = \{g_3\}^{\mathsf{K}}$. But more generally, we show how to rewrite, in a p.r. way, a program computing $\{\bar{e}\}^{\mathsf{J}}$ into one $\{f(\bar{e})\}^{\mathsf{K}}$ computing the same function.

The index e codes a finite ittm programme that syntactically may be run as a programme P_e^{K} or indeed with any other oracle functional K - the grammar of the programme does not impose any conditions on the oracle - it merely asks for values.

This will be done in a uniform manner that is independent of J, K (as long as any other K' under consideration agrees with K on 0 and 1). A query at local time α, $Q^{\mathsf{J}}(e_1, m_1, y_1)$ at level 0, has two phases: it asks if $\{e_1\}^{\mathsf{J}}(m_1, y_1)\downarrow z$ for some z and secondly, if so, it asks for $\mathsf{J}(z)$ which is then returned at local time $\alpha+1$ again at level 0. To effect the translation of this as a recursion in K we slightly modify the mechanism of Exercise 2.9. There the value of $\mathsf{J}(z)$ was introduced by the device of the identity function, as the value returned, following on the subcomputation call $P_\iota(z)$. The original program, P_e, for which this was introduced was modified to $P_{h(e)}$ (for a p.r. h). (And all subcomputation calls $Q^{\mathsf{J}}(e_1, m_1, y_1)$, by recursion, were modified to $P_{h(e_1)}$ *etc.* We here just replace h by $h(e_1,g_3)$ the index function arising in the same way, but replacing the identity query $?Q^{\mathsf{J}}(\iota, z)?$ by the code of the program $\{g_3\}^{\mathsf{K}}(z)$ using z from R. Doing this recursively throughout the computation, yields a new program whose index can be a primitive recursive function of the former, uniformly in g_3; we let this be then $h(e_1,g_3)$. Then $f(e_1) =_{\mathrm{df}} h(e_1,g_3)$ is the function of the theorem. QED

The construction here only depended in trivial ways upon the suitability of K.

Lemma 2.17. $\mathsf{iJ} \leq \mathsf{E}$. *Hence as* E *is trivially recursive in* iJ, *we have* $\mathsf{E} \equiv \mathsf{iJ}$.

Proof: Adapt the methods above. Exercise. QED

Exercise 2.14. Check the last theorem holds under full generality of suitability of K.

Exercise 2.15. Show that there is a p.r. function f so that if H is a functional with $\mathsf{H}(\boldsymbol{m}, \boldsymbol{x}) = \{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x})$, then
$$\{k\}^{\mathsf{H}}(\boldsymbol{m}, \boldsymbol{x}) \simeq \{f(k, e)\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x}).$$

Exercise 2.16. Show that for any functionals I, J, if for any x recursive in I that $\mathsf{I}(x) = \mathsf{J}(x)$, then any relation R on ${}^k\omega$ is semi-recursive (respectively recursive) in I if it is semi-recursive (respectively recursive) in J.

Exercise 2.17. Show that, for suitable K, there are indices p_1, p_2 so that $\{p_1\}^{\mathsf{K}}(e, m, x)\downarrow s$ where s is the least snapshot of $\{e\}(m, x)$ that repeats. And p_2 is such that $?Q^{\mathsf{K}}(p_2, (e, m, y, \langle k, l\rangle))?$ returns 1 if $\{e\}^{\mathsf{K}}(m, y)\downarrow$ and $\{e\}^{\mathsf{K}}(m, y)(k) = l$ and 0 otherwise. In particular a programme P^{K} can compute the graph of convergent $\{e\}^{\mathsf{K}}(m, y)$.

Although we build into our framework, following Kleene, that a query $Q^{\mathsf{K}}(e, m, x)?$ first does some computation, namely $\{e\}^{\mathsf{K}}(m, x)$, and if this is convergent, submits the result to K, actually we can shortcircuit the process, and, if K is suitable, simply query $Q^{\mathsf{K}}(e', (e, m), x)?$ (for some e' which is effectively obtainable from e) and this query returns $1/0$ if $\{e\}^{\mathsf{K}}(m, x)\downarrow/\uparrow$. So our queries can tell us during a computation, convergence/divergence facts. Moreover (see the last Exercise) an ω sequence of queries can tell us the digits of the convergent output of any computation, when it exists.

From this point on we shall assume our functionals are suitable, unless otherwise stated.

Exercise 2.18. Show that there are indices e_0, e_s so that $\{e_0\}^{\mathsf{K}}(e, m, x)\downarrow z^x$, $\{e_s\}^{\mathsf{K}}(e, m, x)\downarrow s^x$, where z^x is the $L[x]$-least code for $L_{\xi^x}[x]$ and s^x is the $L[x]$-least code for $L_{\sigma^x}[x]$, the least level of the $L[x]$ hierarchy with a proper Σ_2-elementary substructure $L_{\xi^x}[x]$.

Lemma 2.18. *There is a p.r. function* f *such that for any* I, $e, k, \boldsymbol{m}, \boldsymbol{x}$
$$\{f(e, k)\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x}) \simeq \{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x}, \lambda n.\{k\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x}))$$
and hence the functions partial recursive in I *are closed under functional substitution.*

Proof: The index $f(e, k)$ is for the procedure that does the following: (A) it first checks for $n = 0$, $1, 2, \ldots$ in turn that $\{k\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x})\downarrow k_n$, for some $k_n \in \omega$, and if so records the value on a scratch tape. If $\{k\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x})\uparrow$ then again the R.H.S. fails to compute anything (as $\lambda n.\{k\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x})$ is not total). Lastly if for some $n\{k\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x})\downarrow y \notin \omega$ then we perform some fixed trivial code with an illfoundedtree to ensure the non-totality of $\lambda n.\{k\}^{\mathsf{I}}(n, \boldsymbol{m}, \boldsymbol{x})$.

Otherwise all is well and we have written some real $z = (k_0, k_1, \ldots) \in {}^{\omega}\omega$ on the scratch tape. (B) We conclude with performing the calculation $\{e\}^{\mathsf{I}}(\boldsymbol{m}, \boldsymbol{x}, z)$. QED

Exercise 2.19. (i) Show that there exists a (ittm-)recursive function H, and a function F partial recursive in H such that F is not partial recursive. (We should not be surprised at this.)

(ii) On the other hand show that for any H recursive in I, and F partial recursive in H that F is partial recursive in I.

Exercise 2.20. (The functional E_1) (i) Define E_1 by:

$$\mathsf{E}_1(x) = \begin{cases} 1 & \text{if } \exists y \forall n\, x(\overline{y \restriction n}) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

where for finite sequences $s \in {}^{<\omega}\omega$, then $\bar{s}$ is a standard recursive number code for s, (0 coding the empty sequence) and for $y \in {}^{\omega}\omega$, $\overline{y \restriction n}$ is then the code of $y \restriction n = \langle y(0), y(1), \cdots, y(n-1)\rangle$. Show that $\mathsf{E}_1 \leq \mathsf{K}$ for any suitable type-2 functional K.

2.3 Ordinal Comparison

For Kleene Recursion the next move would be to prove an Ordinal Comparison Theorem and use this to develop some of the theory of the semi-recursive on I sets. We have potentially two options here along two different axes, to assign ordinals to computations rather as for Kleene recursion, namely as to which stage I^α a computation $\{e\}^{\mathsf{I}}(m,x)$ enters into the monotone inductive definition of all successful computations in I, or else we may look at the overall convergence time $H(\mathsf{I}, e, \boldsymbol{m}, \boldsymbol{x})$ it takes for $\{e\}^{\mathsf{I}}(m,x)$ to converge. They yield somewhat differing prewellorderings on computations. We choose the latter possibility. With the extra power of being able to compute ordinals for lengths of successful computations, some of the results for Kleene Recursion become then simple in this context.

We first note that there are suitable *universal semi-recursive in* I *sets.*

Definition 2.19. $\mathsf{U}^{\mathsf{I}}(e,m,x) \longleftrightarrow \{e\}^{\mathsf{I}}(m,x)\!\downarrow$*;*
$U^{\mathsf{I}}_y(e,m) \longleftrightarrow U^{\mathsf{I}}(e,m,y)$;
$U^{\mathsf{I}}(e,m) \longleftrightarrow U^{\mathsf{I}}(e,m,\bar{0})$.

We say that U^{I} *and* U^{I}_y *are* parametrized by ω*: as the index* $e \in \omega$ *varies we obtain all semi-recursive in* I *(or* I *and* y*) sets.*

Definition 2.20. *For* $\langle e, \langle m, x\rangle\rangle \in \mathsf{U}^{\mathsf{I}}$ *then we set* $|\langle e, \langle m, x\rangle\rangle|^{\mathsf{I}} \simeq$ *the least* $\alpha < H(\mathsf{I}, e, m, x)$ *such that for all* $\beta > \alpha$ *the* OT *of this computation is fixed. We write:* $|\langle e, \langle m, x\rangle\rangle|^{\mathsf{I}}\!\downarrow^{\alpha}$.

This gives a norm on U^{I} and so on any semi-recursive in I set A.

Lemma 2.21. *If* $\mathsf{E} \leq \mathsf{I}$ *then the class of sets semi-recursive in* I *satisfies the Prewellordering Property, and hence forms a Spector class.*

Proof: QED

By virtue of forming a Spector class a number of properties about the semi-decidable sets now follow, most of which we could prove directly for them in any case. But for now, we just list these here with some references to [14].

Corollary 2.22. *The class of sets semi-recursive in* I *has the Reduction property, and its dual class the Separation Property.*

Definition 2.23. *Let* $\Delta^{\mathsf{I}}(x)$ *those sets that are both ittm-semi-recursive-in-*x *and* I *and co-ittm-semi-recursive-in-*x *and* I*. When* $\mathsf{I} = \mathsf{E}$ *we omit it.*

Lemma 2.24. *(cf.* [14], 4D.3) *If* $Q \subseteq {}^{\omega}\omega \times {}^{\omega}\omega$ *is (ittm)-semi-recursive in* E*, then defining*

$$P(x) \longleftrightarrow \exists y \in \Delta\ Q(x,y),$$

we have that P is (ittm)-semi-recursive in E. *By relativisation, if $Q \subseteq {}^{\omega}\omega \times {}^{\omega}\omega \times {}^{\omega}\omega$ is (ittm)-semi-recursive in* E, *then defining*

$$P(x,z) \longleftrightarrow \exists y \in \Delta(z)\ Q(x,z,y),$$

then P is (ittm)-semi-recursive in E.

Lemma 2.25. *(Upper Classification cf.* [14], 4D.14) *The class of x recursive in* E, Δ, *is a semi-recursive in* E *class. Similarly the class of x recursive in z and* E, $\Delta(z)$, *is semi-recursive in* (z, E) *class.*

Lemma 2.26. *(Kreisel Uniformization Theorem cf.* [14], 4B.4) *Suppose $P \subseteq {}^{\omega}\omega \times \omega$ is semi-recursive in* E. *Then P can be uniformized by some $P^* \subseteq P$, with P^* semi-recursive in* E.

And its corollary:

Lemma 2.27. *(Kreisel Δ-Selection Theorem cf.* [14], 4B.5) *Suppose $P \subseteq {}^{\omega}\omega \times \omega$ is semi-recursive in* E. *Assume A is in Δ, and $\forall x \in A \exists n\, P(x,n)$. Then there is a recursive in* E *function $f : {}^{\omega}\omega \longrightarrow \omega$ so that $\forall x \in A\, P(x, f(x))$.*

One might think that there is a *Lower Classification Theorem* to go with the above, namely that the set of reals recursive in I is not recursive in I. However Hinman gives an example in Kleene recursion to show that this is not the case, which we shall repeat here in our ittm setting.

Lemma 2.28. *There exists a functional* I *(such that* $\mathsf{E} \leq \mathsf{I}$*) and $\{x \mid x$ is recursive in* $\mathsf{I}\}$ *is recursive in* I.

Proof: Define I by:

$$\begin{aligned} \mathsf{I}(x) &= \mathsf{E}(x) \qquad \text{if } x \text{ is recursive in } \mathsf{I} \\ &= \mathsf{E}(x) + 2 \text{ otherwise.} \end{aligned}$$

Since $\mathsf{E}(x) = 0 \longleftrightarrow \mathsf{I}(x) = 0$ or 2, clearly $\mathsf{E} \leq \mathsf{I}$ (which is in any case trivial in our setting). By the specification of I and Exercise 2.16:

$$P =_{\mathrm{df}} \{x \mid x \text{ is recursive in } \mathsf{I}\} = \{x \mid x \text{ is recursive in } \mathsf{E}\}.$$

Now since $x \in P \longleftrightarrow \mathsf{I}(x) < 2$, P is recursive in I. QED

Exercise 2.21. Show that $\{x \in {}^{\omega}\omega \mid x$ is recursive in $\mathsf{E}\}$ is recursive in E.

Definition 2.29. *We let $|-|_0^{\mathsf{I}}$ be the regular norm on ω^2 induced by restricting the prewellordering $\preceq$ on U^{I} to sequences of type $(e, m, \bar{0})$ ($\bar{0}$ is the zero function); thus*

$$|e,m|_0^{\mathsf{I}} = \sup^+ \left\{ |g,n|_0^{\mathsf{I}} : (g,n,\bar{0}) \prec (e,m,\bar{0}) \right\}.$$

Let $\kappa^{\mathsf{I}} = \sup^+ \left\{ |e,m|_0^{\mathsf{I}} : U^{\mathsf{I}}(e,m) \right\}$.

Definition 2.30. *For* I *a functional,let α_0^{I} be the least ordinal not the order type of a wellordering of ω which is recursive in* I.

For I then, α_0^{I} is the analogue of $\omega_1^{\mathsf{I},\mathrm{ck}}$.

Lemma 2.31. $\kappa^{\mathsf{I}} = \alpha_0^{\mathsf{I}}$.

Proof: [To come] QED

In particular the following question is pertinent for the simplest functionals:

Question: What is $\alpha_0 =_{\rm df} \alpha_0^{\mathsf{E}}$?

By our constructions to date, we have seen that it must be much larger than ζ (see Ex.2.18). It is part of our task to identify this ordinal.

Exercise 2.22. Let x be ittm-recursive in I. Show that $\zeta^x < \alpha_0^{\mathsf{I}}$. Conversely for any $\tau < \alpha_0^{\mathsf{I}}$ there is y ittm-recursive in I with $\tau < \zeta^y$. Thus α_0^{I} is a limit of extendibles.

[The next two have not been checked in general.]

Lemma 2.32. *(Boundedness Theorem) If A is co-semi-recursive in I and $A \subseteq U^{\mathsf{I}}$ then*
$\sup^+ \{|u|^{\mathsf{I}} : u \in A\} < \kappa^{\mathsf{I}}$.

Lemma 2.33. *(Hierarchy Theorem) Let $U^{\mathsf{I}}_\tau =_{\rm df} \{u : u \in U^{\mathsf{I}} \wedge |u|^{\mathsf{I}} < \tau\}$. For any $R \subseteq {}^k\omega$,*
R is recursive in I iff $\exists \tau < \kappa^{\mathsf{I}} \left(R \text{ is many-one reducible to } U^{\mathsf{I}}_\tau\right)$
iff $\exists \tau < \kappa^{\mathsf{I}} \left(R = \{\boldsymbol{m} : \langle e, \boldsymbol{m}\rangle \in U^{\mathsf{I}}_\tau\}\right)$

Lemma 2.34. *(Gandy Selection Theorem) There exists a functional $\mathrm{Sel}^{\mathsf{I}}$ partial recursive in I so that for any e, m, y the following are equivalent:*

(i) $\exists g\, (\{e\}^{\mathsf{I}}(g, m, y)\downarrow)$
(ii) $\{e\}^{\mathsf{I}}(\mathrm{Sel}^{\mathsf{I}}(e, m, y), m, y)\downarrow$.

Proof: Exercise. QED

3 Computation Lengths

As with the basic ittm's the strength of the model is tied up with the length of computations possible or needed on or by the model; in turn that is mutually tied up with the class of reals so recursive (the slogan "clockables are writables" is apposite). The same considerations are true of the generalised ittm recursions here. Even for recursions in E we see (Exercise 2.18) there are computations recursive in E that on input x compute (a code for) $L_{\Sigma^x}[x]$, or for $L_{\zeta^x}[x]$. Combining this with a programme like the Theory Machine we shall see that much longer sections of the L-hierarchy can be computed on integer input, indeed done in the simplest fashion such a programme would loop at the first extendible ordinal that is a limit of such. So we first investigate this hierarchy and then see how to compute through long initial segments.

3.1 Extendability Hierarchy

For E a class of ordinals, let E^* denote the class of its limit points.

Definition 3.1. *We classify Σ_2-extendible ordinals as follows. Define by recursion on α the class E^α of α(Σ_2)-extendible ordinals:*

$$\begin{aligned} E^0 &= \{{}^0\zeta \mid {}^0\zeta \text{ is } \Sigma_2\text{-extendible}\}; \\ E^{\alpha+1} &= \{{}^{\alpha+1}\zeta \mid {}^{\alpha+1}\zeta \in (E^\alpha)^* \cap E^0\}; \\ E^\lambda &= \bigcap_{\alpha<\lambda} (E^\alpha)^* \cap E^0 \text{ for } \mathrm{Lim}(\lambda). \end{aligned}$$

Here we decorate the variable ζ with the prefix indicating its minimum level of extendibility. We shall let ${}^{\alpha}\Sigma$ indicate that for some ${}^{\alpha}\zeta$, $({}^{\alpha}\zeta, {}^{\alpha}\Sigma)$ is a α-extendible pair (we often be interested in finite $\alpha = k$). Obviously for any γ the least element of E^k greater than γ is always an element of $E^k \setminus E^{k+1}$, *i.e.* is k-extendible but not $k+1$-extendible. We extend the definition, relativising to reals x the notion of α-x-extendibles, $E^{\alpha}[x]$, where an x-extendible pair is a ξ, σ with $L_{\xi}[x] \prec_{\Sigma_2} L_{\sigma}[x]$ *etc.*.

Notice that for any ${}^1\zeta$ if $L_{{}^1\Sigma}$ is the natural Σ_2-end extension of $L_{{}^1\zeta}$, then ${}^1\Sigma$ is also in $(E^0)^*$ (but not necessarily in E^1). This holds by simple Π_2-reflection of the statement that there are arbitrarily large elements of E^0 below ${}^1\zeta$ up to ${}^1\Sigma$. So in fact there must be Σ_2-extendible pairs of the form $({}^0\zeta, {}^0\Sigma)$ which are *nested* in the interval $({}^1\zeta, {}^1\Sigma)$, that is ${}^1\zeta < {}^0\zeta < {}^0\Sigma < {}^1\Sigma$. This is suggestive of the kind of linearized computation that allows one depth of subroutine call, thus the top node of its computation tree has rank only 1. However the computation continues sufficiently far that it only enters a final loop at ${}^1\zeta$, all the while making subroutine calls to nodes ν_{α}, (all at $\Lambda = 1$) for α unboundedly in ${}^1\zeta$, and thence by reflection it must be doing so for α unboundedly in ${}^1\Sigma$. But at ${}^1\Sigma$ it drops back to ${}^1\zeta$.

Lemma 3.2. *Let $P_e^{\mathsf{E}}(k)$ be such that no query $Q^{\mathsf{E}}(e_1, m_1, y_1)$ is made so that $\{e_1\}^{\mathsf{E}}(m_1, y_1)$ itself invokes a query (in other words the rank of $\mathfrak{T}(e,k) = 1$). Then if ${}^1\zeta$ is the least element of E^1 $P_e^{\mathsf{E}}(k)$ enters a final loop by stage ${}^1\zeta$.*

This picture propagates: if queries are made to a greater depth in calculations of $P_e^{\mathsf{E}}(n)$ a greater rank of Σ_2-extendibles may be needed to represent the ordinal length of time for the overall computation, with in turn, the rank corresponding to the rank of the computation tree $\mathfrak{T}(e,n)$.

Lemma 3.3. *Suppose $P_e^{\mathsf{E}}(k)\!\downarrow z$. Let $\rho = \rho^{\mathsf{E}}(\langle\langle e,k\rangle, z\rangle$. If ${}^{\rho}\zeta$ is the least element of E^{ρ}, then $P_e^{\mathsf{E}}(k)$ enters a final loop by at latest stage ${}^{\rho}\zeta$. More generally, if $P_e^{\mathsf{E}}(k,x)\!\downarrow z$, $\rho = \rho^{\mathsf{E}}(\langle\langle e,k,x\rangle, z\rangle$ and ${}^{\rho}\zeta$ is the least element of $E^{\rho}[x]$, then $P_e^{\mathsf{E}}(k,x)$ enters a final loop by stage ${}^{\rho}\zeta$.*

Proof: By induction on ρ. Suppose the Lemma holds for $\rho' < \rho = \rho^{\mathsf{E}}(\langle\langle e,k\rangle, z\rangle$ as specified.

Claim: $\Lambda({}^{\rho}\zeta) = \Lambda({}^{\rho}\Sigma) = 0$.

Proof: Suppose not, and $\Lambda({}^{\rho}\zeta) = \Lambda({}^{\rho}\Sigma) = k > 0$. As $\Lambda({}^{\rho}\zeta) > 0$ there is some 'current query' $Q(e_i, m_i, x_i)$ in process at stage ${}^{\rho}\zeta$. We note that this subcomputation could not have been called at stage ${}^{\rho}\zeta$ itself, but must have been called at an earlier stage $\alpha_0 < {}^{\rho}\zeta$. (For otherwise, ${}^{\rho}\zeta$ would be Σ_2-definable in $L_{{}^{\rho}\Sigma}$, as the last point at which a subcomputation at this level was invoked - because if the query $Q(e_i, m_i, x_i)$ was completed, control would have passed up to level $k-1$. But then by Σ_1 reflection that would happen unboundedly in ${}^{\rho}\zeta$, and thence $\Lambda({}^{\rho}\zeta) < k$ - a contradiction.) By similar reasoning we have that this subcomputation is run at this level at all stages $\beta \in [\alpha_0, {}^{\rho}\Sigma)$ - again it could not be completed at a stage $\tau < {}^{\rho}\Sigma$, as then control would pass to level $k-1$, and result in a contradiction once more.

We next note that if $\{e_i\}(m_i, x_i)\!\downarrow w_i$, then $\rho' := \rho^{\mathsf{E}}(\langle\langle e_i, m_i, x_i\rangle, w_i\rangle) < \rho$ (being a subcomputation of $P_e^{\mathsf{E}}(k)$). However $x_i \in L_{\rho_{\zeta}}$ (as the query $Q(e_i, m_i, x_i)$ was invoked at stage $\alpha_0 < {}^{\rho}\zeta$). Moreover ${}^{\rho}\zeta$ is a limit of x_i-${}^{\rho'}\zeta$-extendibles. Let (ξ, σ) be such a pair, with $\alpha_0 < \xi < \sigma < {}^{\rho}\zeta$. By the (more general case of the) inductive hypothesis $\{e_i\}(m_i, x_i)$ has (ξ, σ) as a looping pair, and consequently the query $Q(e_i, m_i, x_i)$ is completed by stage σ. But then $\Lambda(\sigma + 1) = k - 1$, contradicting it being constantly k in $[\alpha_0, {}^{\rho}\Sigma)$. Contradiction! QED Claim

Hence the *Claim* holds: but this is the Lemma: the snapshots of the master computation $P_e^{\mathsf{E}}(k)$ at times ${}^{\rho}\zeta$ and ${}^{\rho}\Sigma$ are identical. QED Lemma

The methods above always allow us to calculate the k-extendibles (and even α-extendibles) above any ordinal, as we shall now turn to.

We first collect together some of the above Facts and results, in order to abbreviate our descriptions of algorithms. This will help to have a library of basic algorithms which we shall simply quote as being 'recursive in K' without further justification. (we just use the adjectivd 'basic' to classify them; we are not intending that they form a basis for any class.)

Definition 3.4. *(Basic Computations-*BC*) Let K be suitable.*

(i) Any standard ittm-computation $P_e(n,x)$ is Basic.

(ii) If a code y for an α ordinal is given, then the computations that compute: for any x (a code for) $L_\alpha[x]$ and the satisfaction relation for $L_\alpha[x]$ are both Basic (in x,y). (These computations show those objects are K-recursive, if α is).

The following are all K-recursive, and Basic:

(iii) The function $x \rightarrowtail \tilde{x}$; and thus $x \rightarrowtail T^2_{\zeta^x}$ (cf. Ex. 2.18)

(iv) The function that computes $x \rightarrowtail \zeta(x)$, the least x-Σ_2-extendible;

(v) The function that computes $x \rightarrowtail \Sigma(x)$, the larger of the next extendible pair in x;

(vi) The function that computes $x \rightarrowtail \Sigma(x)^+$ (the next admissible beyond $\Sigma(x)$).

Stronger ordinals than simply $\Sigma(x)^+$ can be K-recursive:

Lemma 3.5. *There is a (Turing) recursive sequence of indices $\langle e_i | i<\omega\rangle$ so that for any $\alpha<\omega_1$ with a code $x\in 2^{\mathbb{N}}$, $P^{\mathsf{K}}_{e_i}(x)$, with $\mathrm{rk}(\mathfrak{T}^{\mathsf{K}}(e_i,x))=i$, computes a code for the next i-x-extendible ${}^i\zeta>\alpha$.*

Proof: For $i=0$ this has been done using Basic Computations. Suppose e_i has been defined, and we describe the programme $P^{\mathsf{K}}_{e_{i+1}}$. Assume without loss of generality that $\alpha=0$, $x=\mathrm{const}_0$. Then $P^{\mathsf{K}}_{e_i}(0)$ computes a code for the least i-extendible, $\zeta_0:={}^i\zeta$ say. By a basic computation let a slice of the scratch tape R be designated to hold $T^2_{\zeta_0}$; $R:=T^2_{\zeta_0}$. A code for ζ_0, W_{ζ_0} say, is recursive in $T^2_{\zeta_0}$. Now compute $P^{\mathsf{K}}_{e_i}(W_{\zeta_0})$. This yields the next i-extendible $\zeta_1:={}^i\zeta_1$. Now, using Basic Computations, write successively to R the theories $T^2_{\zeta_0}, T^2_{\zeta_0+1},\ldots,T^2_{\zeta_0+\beta},\ldots$ for $\beta<\zeta_1$. We note that at limit stages $\lambda\leq\zeta_1$, R will contain "liminf" theories $\hat{T}_\lambda=\mathrm{Liminf}_{\alpha\to\lambda}T^2_\alpha$ (by the usual automatic ittm liminf process) but that T^2_λ is uniformly r.e. in $\hat{T}_\lambda$ as we saw above. And again a code W_λ for λ is then arithmetic in T^2_λ - uniformly in λ. The point of this exercise of writing theories to R is to ensure continuability of the computation, and that we do not start to loop too early. (Another way to put this is to say that it ensures sufficient 'universality'.) The writing out of all levels of the theories to R is a precautionary step: in general we do not have $\hat{T}_{i+1\zeta}=\mathrm{liminf}_{i\zeta\to i+1\zeta}\hat{T}_{i\zeta}$. However $\hat{T}_{i+1\zeta}\big(=T^2_{i+1\zeta}\big)$ is what we shall need to calculate ${}^{i+1}\zeta$.

Set $R:=T^2_{\zeta_1}\in L_{\zeta_1+1}$; now compute $P^{\mathsf{K}}_{e_i}(W_{\zeta_1})$ and repeat this process. As there is no means for the process we are describing to halt, there is a least looping pair of ordinals for it, (ζ,Σ) say. Let $({}^{i+1}\zeta,{}^{i+1}\Sigma)$ be the least $i+1$-extendible pair. We claim that this is the pair (ζ,Σ). Suppose $\zeta<{}^{i+1}\zeta$. By the repetition of the contents of R in the loop points, we have $\hat{T}_\zeta=\hat{T}_\Sigma$ in the above algorithm, hence $T^2_\zeta=T^2_\Sigma$, and thus:

Claim:$L_\zeta\prec_{\Sigma_2}L_\Sigma$.

Proof: of Claim: (Sketch) Notice that ζ is least with $T^2_\zeta=T^2_\Sigma$ (otherwise there'd be an earlier beginning of our loop). Then the Σ_2-skolem hull of $\varnothing$ in L_Σ is L_ζ. QED

But then ζ is an extendible limit of i-extendibles, as ζ is a limit point of this looping process. This contradicts the minimality of ${}^{i+1}\zeta$. Hence ζ equals the latter, and $\Sigma = {}^{i+1}\Sigma$ follows.

Hence we may compute $\hat{T}_{i+1\zeta} = T^2_{i+1\zeta}$, as the eventually fixed output by means of the above looping procedure (and determining this convergent output requires the extra $+1$ depth to $i+1$ to the overall calculation). It is thus recursive in K (and x). We let $P^{\mathsf{K}}_{e_{i+1}}$ be the programme of the procedure just described followed by the basic computation that finds a code $W_{i+1\zeta}$ for ${}^{i+1}\zeta$ by a method uniformly arithmetic in the now K-recursive $T^2_{i+1\zeta}$.

Finally note that the continuing description of the programme $P^{\mathsf{K}}_{e_{i+2}}$ from $P^{\mathsf{K}}_{e_{i+1}}$ merely repeats the above but altering only a few suffices. We may thus determine a recursive function $i \mapsto e_{i+1}$. □

Lemma 3.6. *Suppose $\{e\}^{\mathsf{E}}(\boldsymbol{m}, \boldsymbol{x})$ is any recursion effected by the program P^{E}_0, that, in addition to what else it does, in a register R on its scratch tape writes, for each step ξ of the calculation it takes, the theory $T^2_\xi[\boldsymbol{x}] = \Sigma_2\text{-Th}(L_\xi[\boldsymbol{x}], \in)$. Suppose P^{E}_0 has as least pair of looping ordinals $\zeta_0 < \Sigma_0$. Then, uniformly in e we may find e' so that $\{e'\}^{\mathsf{E}}(\boldsymbol{m}, \boldsymbol{x})$, computes a code for Σ_0.*

Proof: We set $\boldsymbol{x} = 0$. Note that as $P_0 = P^{\mathsf{E}}_0$ computes $\hat{T}_{\zeta_0} = \hat{T}_{\Sigma_0}$ in its algorithm, and such are part of its looping cycle, we have that $T^2_{\zeta_0} = T^2_{\Sigma_0}$ and so $L_{\zeta_0} \prec_{\Sigma_2} L_{\Sigma_0}$. (Again for the latter, the Σ_2-Skolem hull of $\varnothing$ in L_{Σ_0} must be L_{ζ_0} for otherwise it would be some $\bar{\zeta} < \zeta_0$; but then P^{E}_0 is looping between $\bar{\zeta}$ and ζ_0.) We may write a program $P = P^{\mathsf{E}}$ so that using appeals to the E functional, we ask ω-many questions as to whether $\ulcorner \varphi \urcorner \in \hat{T}_{\zeta_0}$ for a simulated computation of P_0, and hence can write out $\hat{T}_{\zeta_0}$ on a scratch tape. For limit stages $\lambda \leq \Sigma_0$, uniformly r.e. in $\hat{T}_\lambda$ is T^2_λ. After P has done this, it reruns a simulation of P_0, waits until the latter has written $\hat{T}_{\zeta_0}$ (which P can now recognize). After this has happened it waits until the least later limit stage, which we shall call α_n, with $T^2_{\alpha_n} \cap n = T^2_{\zeta_0} \cap n$. (One can argue that such α_n must occur unboundedly in Σ_0, as eventually we have $T^2_{\Sigma_0} = T^2_{\zeta_0}$ is in P_0's register.) When this occurs, then in a uniform arithmetical fashion in $T^2_{\alpha_n}$ we can compute the code W_{α_n} - the L-least code for α_n. This can be done for increasing n, resulting in a sequence of codes $\langle W_{\alpha_n} \rangle_n$ that P may write out. The point is that $\sup_n \alpha_n = \Sigma_0$ (as Σ_0 is the least $\beta > \zeta_0$ with $T^2_\beta = T^2_{\zeta_0}$). Then from this sequence of codes, P proceeds to assemble one for Σ_0. □

Similar arguments to the latter allied with Lemma 3.5 show:

Lemma 3.7. *There is a recursive sequence of indices $\langle e'_i | \, i < \omega \rangle$ so that $P^{\mathsf{K}}_{e'_i}(\boldsymbol{m}, \boldsymbol{x})$ writes a code for $L_{i_{\Sigma(\boldsymbol{x})}}[\boldsymbol{x}]$, the least Σ_2-extension of $L_{i_\zeta}[\boldsymbol{x}]$ where $({}^i\zeta, {}^i\Sigma)$ is the least i-$\boldsymbol{x}$-extendible pair in $E^i(\boldsymbol{x})$.*

The last lemma shows only that we can recursively find, for example, the least Σ_2-extendible in a real x, namely ζ^x. However more is possible: given (e, m, x) we may, recursively in K, compute a code for σ_0 where (ζ_0, σ_0) is the least looping pair of ordinals for the computation $\{e\}^{\mathsf{K}}(m, x)$ (assuming of course the latter has a wellfounded tree $\mathfrak{T}^{\mathsf{K}}(e, m, x)$). In fact this gives us the basis of a version of a *universal Kleene T-predicate*. Recall that in ordinary Turing recursion, there is a recursive (in fact p.r.) predicate T so that $\{e\}(m)\!\downarrow k \longleftrightarrow \exists u \in \mathrm{Seq}\, T(e, m, u) \wedge \mathrm{last}(u) = k$. The sequence u codes the whole (finite) course of computation on integers. For us a course of computation is a transfinite wellfounded sequence of snapshots which when convergent delivers a real. Thus our $\mathbb{T}$-Predicate Theorem will read as follows:

Theorem 3.8. *There is a Δ^1_1-predicate $\mathbb{T}$ so that*
$\{e\}^{\mathsf{K}}(m, x)\!\downarrow y \longleftrightarrow \exists u \in \mathrm{WF}\, \mathbb{T}(\mathsf{K}, e, m, u) \wedge \mathrm{last}(u) = y$.

[XXX The above too soon??XXX]

Again u will code a wellfounded sequence of snapshots of the whole computation tree $\mathfrak{T}^{\mathsf{K}}(e, m, x)$ associated to $\{e\}^{\mathsf{K}}(m, x)$, viewed as a sequential process. For this to work we, as a minimum, need to be able to compute (recursively in terms of K, m, x) the ordinal $H(\mathsf{K}, e, m, x)$ which is the overall length of this computation. To put it in terms often used for ordinary ittm's, the 'clockable' ordinal $H(\mathsf{K}, e, m, x)$ needs to be 'writable', which is what the next Lemma asserts. We stick with E.

Exercise 3.1. There is a p.r. function g_0 so that if $\{e\}^{\mathsf{E}}(m, x)$ makes a query $?Q^{\mathsf{E}}(e_1, m_1, x_1)?$ whilst if $\{e_1\}^{\mathsf{E}}(m_1, x_1)\downarrow w_1$ then $\{g_0(e)\}^{\mathsf{E}}(m, x)$ runs as $\{e\}^{\mathsf{E}}(m, x)$ but there is a reserved infinite slice R of scratch tape of the computation $\{g_0(e)\}^{\mathsf{E}}(m, x)$ so that not just $\mathsf{E}(w_1)$ is returned following the query but all of w_1 is returned by being written to R. [Hint: Just enlarge $\{e_0\}$ by adding to each $?Q^{\mathsf{E}}(e_1, m_1, x_1)?$ an infinite sequence of queries of the form $?Q^{\mathsf{E}}(f, n, e_1, m_1, x_1)?$ where $\{f\}^{\mathsf{E}}(n, e_1, m_1, x_1)\downarrow 1/0 \longleftrightarrow w(n) = 1/0$.]

Exercise 3.2. Show that there is a p.r. g_1, so that considering g_0 from the previous Exercise, if whenever a query $?Q^{\mathsf{E}}(e_1, m_1, x_1)?$ in $\{e\}^{\mathsf{E}}(m, x)$ results in $\{e_i\}^{\mathsf{E}}(m_i, x_i)\downarrow w_i$ with w_i being returned to R, then w_i is tested to see if it is in WO. If so then, if $w_i = \alpha_i$, then we record the increasing sum $\Sigma\alpha_i$ with $\{g_1(e)\}^{\mathsf{E}}(m, x)\downarrow W$ with $\|W\| = \Sigma\alpha_i$, the eventual sum over all such ordinals.

Lemma 3.9. *There is a p.r. function k so that if $\{e\}^{\mathsf{E}}(m, x)$ has a wellfounded computation tree $\mathfrak{T}^{\mathsf{E}}(e, m, x)$ then $\{k(e)\}^{\mathsf{E}}(m, x)$ computes a code for $H(\mathsf{E}, e, m, x)$.*

Remark: Note that we do not need $\{e\}^{\mathsf{E}}(m, x)\downarrow$ to assert this. $H(\mathsf{E}, e, m, x)$ is defined even if $\{e\}^{\mathsf{E}}(m, x)\uparrow$, *i.e.*, as long as $\mathfrak{T}^{\mathsf{E}}(e, m, x)$ is wellfounded.

Proof: [Under construction.]

The function k will be defined by making use of the recursion theorem. Let us suppose the computation whose length function $H = H(\mathsf{E}, e_0, m_0, x_0)$ to be calculated is $\{e_0\}^{\mathsf{E}}(m_0, x_0)$ at level $\Lambda = 0$. Our aim will be to calculate $H(\mathsf{E}, e_0, m_0, x_0)$ in terms of the lengths $H(\mathsf{E}, e_i, m_i, x_i)$ of the query calls $?Q(e_i, m_i, x_i)?$ that occur during $\{e_0\}^{\mathsf{E}}(m_0, x_0)$, and that take place on those nodes ν_i on level $\Lambda = 1$. We use the recursion theorem to define a suitable function H and then show by induction on ranks of computation that all such $H(\mathsf{E}, e_i, m_i, x_i)$ have been correctly computed.

Let $F(g, e_0, m_0, x_0)$ be the generalised ittm recusive function defined as follows. We regard this as dependent uniformly on the index g, and so we think of F as $\{f(g)\}(e_0, m_0, x_0)$

(1) F first runs a simulation, or copy, of $\{e_0\}^{\mathsf{E}}(m_0, x_0)$ on some set-aside work tape. From this it harvests its final looping snapshot s to some 'register' A of work tape. (This is done in the manner of Exercise 2.10) With this done:

(2) It runs again a copy of $\{e_0\}^{\mathsf{E}}(m_0, x_0)$; for each top level step here of this copy we set its τ'th snapshot to be $s(\tau)$; thus the repeating snapshot s in (1) is $s(\xi)$ for some ξ which will be repeated as $s(\delta)$ for some later δ. We shall just consider the case that $\mathrm{Lim}(\xi)$, and $\mathrm{Lim}(\delta)$ - as the other cases the reader can adapt from what follows. This time, additionally for each top level $\Lambda = 0$ step in this calculation it takes, it extends the code for a copy of some stage of the $L[x_0]$-hierarchy it has on a reserved workspace, $L_{\alpha(\tau)}[x_0]$ say, together with its Σ_2-theory $T^2_{\alpha(\tau)}$, to that of $L_{\alpha(\tau+1)}[x_0]$ and $T^2_{\alpha(\tau+1)}$, where at this successor step $\alpha(\tau + 1) = \alpha(\tau) + 1$. (We may drop the x_0 now for brevity.) We suppose this code and theory are continually written in a register B.

(3) If in the course of running this copy it encounters at stage some τ (that is with $s(\tau)$ as the current snapshot) a query $?Q(e_i, m_i, x_i)?$ (which we'll abbreviate as Q_i):

(i) it adds one more level to the code and theory in B - as at (2);

(ii) instead of acting directly on the query Q_i the simulation instead performs $\bar{Q}$ an infinite chain of queries $?Q(\bar{g}(g), e_i, m_i, x_i, n)?$ for $n < \omega$ which has the effect of

a) writing $\mathsf{E}(y_i)$ to the appropriate part of (the simulated) $\{e_0\}^{\mathsf{E}}(m_0, x_0)$'s work tape, if $\{e_i\}^{\mathsf{E}}(m_i, x_i)\downarrow y_i$, and this determines the snapshot $s(\tau + 1)$ of the simulation;

b) then this is followed by the writing of the n'th digits of z_i to a predetermined register, C say, of F's worktape where $\{g\}^{\mathsf{E}}(e_i, m_i, x_i)\downarrow z_i$ if this computation is convergent. (This is then done in the manner of Exercise 2.9 which showed us how to adjust a single query into an ω-sequence of queries, to return the whole of the real on the output of a stabilized output tape.)

(4) If $\bar{Q}$ is completed, z_i, now on C, is tested for being a code of an ordinal or not; if so, and its length is η say, then F adds η levels to the codes of the $L[x_0]$ hierarchy in the register B, thus setting $\alpha(\tau+1)=\alpha(\tau)+\eta$, and runs also through the next η levels of their theories. These then look like some $L_{\alpha(\tau)+\eta}[x_0]$ and $\hat{T}^2_{\alpha(\tau)+\eta}[x_0]$. F then returns to (2) and continues the simulation there, noting that at (3)(ii)a) the value $\mathsf{E}(y_i)$ was written to the appropriate place.

(5) If at some stage ξ say of the simulation, the computation tape of $\{e_0\}^{\mathsf{E}}(m_0,x_0)$ has the same snapshot $s=s(\xi)$ as the repeating snapshot stored in A, this fact is taken note of, and the current theory in B, $\hat{T}^2_{\alpha(\xi)}[x_0]$ say, is written and kept in a register B_0. (Note this may be a liminf theory, if ξ is a limit - as may usually be the case.) The computation F continues as before running through (2) $\longrightarrow$ (4) $\longrightarrow$(2) until the snapshot s reappears for a second time, say that $s=s(\delta)$ also. (Or just stays with (2) if no further queries are reached.) At this point F looks in B and sees some $\hat{T}^2_{\alpha(\delta)}[x_0]$ say, and runs a check on it: if this is different from the theory $\hat{T}^2_{\alpha(\xi)}[x_0]$ in B_0, then a code $w_{\alpha(\delta)}$ for the ordinal $\alpha(\delta)$, which is r.e. in $\hat{T}^2_{\alpha(\delta)}[x_0]$, is written as its final output of F to its OT. If not, and these two theories are the same, then we have that that part of the F mechanism that is writing to register B is resulting in a looping action on these theories.

In order to get a code for $\alpha(\delta)$, and not for the $\alpha(\xi)$ that is r.e. in this theory $T(\alpha(\delta))=_{\mathrm{df}}\hat{T}^2_{\alpha(\delta)}[x_0]$, (which by the case hypothesis asserts that $T(\alpha(\delta))=T(\alpha(\xi))=_{\mathrm{df}}\hat{T}^2_{\alpha(\xi)}[x_0]$) we do the trick of the proof of Lemma 3.6 to calculate from $\alpha(\xi)$ and $T(\alpha(\xi))$ the ordinal $\alpha(\delta)$ as the supremum of the ordinals $\langle\alpha_n\rangle_n$ where an increasing (in finite size) number of cells are fixed and/or have changed value. We use $L_{\alpha(\xi)}[x_0] \prec_{\Sigma_2} L_{\alpha(\delta)}[x_0]$ in order to do this. (Check: they have the same Σ_2-theories and then $L_{\alpha(\xi)}[x_0]$ is the range of the Σ_2-skolem function for $L_{\alpha(\delta)}[x_0]$.) This delivers for us a code $w_{\alpha(\delta)}$ for $\alpha(\delta)$, which is then output by F.

This completes the description of F. By the Recursion Theorem there is g_0 so that $F(g_0,\, e_0,\, m_0,\, x_0) \simeq \{g_0\}(e_0,\, m_0,\, x_0)$. The intention is then that our sought after $H(\mathsf{E},\, e_0,\, m_0,\, x_0)$ is $\{g_0\}(e_0,\, m_0,\, x_0)$. We prove this in general by induction on the ranks $\rho^{\mathsf{E}}(\langle\langle e,\, m,\, x\rangle,\, z\rangle)$ of convergent computations.

Note that if any of the sub-computations $\{g_0\}^{\mathsf{E}}(e_i,\, m_i,\, x_i)$ fails to converge (and so *a fortiori* $\{\bar{g}(g_0)\}(e_i,m_i,x_i,n)$ fails to converge - for any n), then $F(g_0,e_0,m_0,x_0)$ fails to converge.

So suppose that $H(\mathsf{E},\, e_i,\, m_i,\, x_i) = \{g_0\}(e_i,\, m_i,\, x_i)$ has been proven for all $\langle\langle e_i,\, m_i,\, x_i\rangle,\, z_i\rangle$ with $\rho^{\mathsf{E}}(\langle\langle e_i,\, m_i,\, x_i\rangle,\, z_i\rangle) < \rho^{\mathsf{E}}(\langle\langle e_0,\, m_0,\, x_0\rangle,\, z_0\rangle)$. This means that the digits of the ordinal coded as z_i that are handed up to the worktape of $F(g_0,\, e_0,\, m_0,\, x_0)$ as the answers to the query $?Q(\bar{g}(g_0),e_i,m_i,x_i,n)?$ are precisely those of a code for $\eta=H(\mathsf{E},e_i,m_i,x_i)$ (by the induction hypothesis). This is the ordinal η added to the lengths of the codes and theories of the $L[x_0]$ hierarchy that are being built in B. Thus what is being added is precisely that ordinal needed to continue to construe the running of $\{e_0\}^{\mathsf{E}}(m_0,\, x_0)$ in a *linear* fashion up to this point, that is achieved by adding on the ordinals for $\{e_i\}^{\mathsf{E}}(m_i,\, x_i)$. What then could possibly go wrong? It is that the liminf theory $\hat{T}^2_{\alpha(\delta)}[x_0]$ arrived at in the notation of (5) in the description of $F(g_0,\, e_0,\, m_0,\, x_0)$ has itself occurred earlier, in our computational mechanism F, and that for some $\alpha(\xi) < \alpha(\delta)$ we have $\hat{T}^2_{\alpha(\delta)}[x_0]=\hat{T}^2_{\alpha(\xi)}[x_0]$. The ordinal output of F encoded on B would then be the beginning of a loop in the calculation of theories and instead a code for the earlier ordinal $\alpha(\xi)$ is output by F. However we have catered for exactly this eventuality by having F run a check on $\hat{T}^2_{\alpha(\delta)}[x_0]$, and if necessary calculating $\alpha(\delta)$, and not $\alpha(\xi)$, as its output.

Claim: $H(\mathsf{E},e_0,m_0,x_0)=\{g_0\}(e_0,m_0,x_0)$.

Proof: The recursion above keeps changing theories, and adding on ordinals, according to the definition of $H(\mathsf{E},\, e_0,\, m_0,\, x_0)$: it is the linear sum total of the previous processes, as expressed in Definition 2.6. This is clear as long as the addition of theories keeps extending the hierarchy through the liminfs, and not dropping back. We then just check that this dropping back can only first occur at the right place.

Subclaim: Suppose $T(\alpha(\lambda')) = T(\alpha(\lambda))$ *for some* $\lambda' < \lambda$ *for some lexicographically least pair* $(\lambda,\, \lambda')$. *Then* $\lambda' = \xi$ *and* $\lambda = \delta$, *where* $(\xi,\, \delta)$ *is the pair of ordinals for the least finally looping snapshots* $s(\xi) = s(\delta)$ *in the run of* $\{e_0\}^{\mathsf{E}}(m_0, x_0)$.
Proof: Let the supposition hold as stated and suppose that we have λ', λ as the least pair for which this happens. As the two theories are the same we have that $L_{\alpha(\lambda')}[x_0] \prec_{\Sigma_2} L_{\alpha(\lambda)}[x_0]$. (Check: they have the same Σ_2-theories and $L_{\alpha(\lambda')}[x_0]$ is the range of the Σ_2-skolem function for $L_{\alpha(\lambda)}[x_0]$.)

The point is that now the snapshot sequences $\langle s(\gamma)|\ \gamma < \delta\rangle$ are Δ_1-definable over $L_{\alpha(\lambda')}[x_0]$ (respectively $L_{\alpha(\lambda)}[x_0]$) for any $\gamma < \alpha(\lambda')$ $(< \alpha(\lambda)$ respectively), because the ordinals $\alpha(\lambda')$ and $\alpha(\lambda)$ are long enough to run the linearized computations inside those models. (One might have expected to see "$\gamma < \lambda'$ $(< \lambda$ respectively))" written here, being the order type of the stages (2) $\longrightarrow$(4) cycled through above but note that $\lambda' = \alpha(\lambda')$ because otherwise we could define $\alpha(\gamma)$ as a Σ_1-definable function cofinal in $\alpha(\lambda')$ for $\gamma < \lambda'$. This would contradict the admissibility of $L_{\alpha(\lambda')}[x_0]$. Hence $\lambda = \alpha(\lambda)$ also by Σ_2-reflection.)

Moreover then $s(\alpha(\lambda'))$ is $\Sigma_2(L_{\alpha(\lambda')}[x_0])$ and this equals $s(\alpha(\lambda))$ by Σ_2-reflection. But then these two are repeating snapshots. By the usual arguments they can only be $s(\xi)$ and $s(\delta)$.
QED Subclaim

We thus can only have a drop-back of theories in the computation of $\{g_0\}$ at stage δ. But then $\{g_0\}$ is arranged to output a code $w_{\alpha(\delta)}$ for $\alpha(\delta)$ in this case. QED *Claim and Lemma*

Theorem 3.8 can be established by considering a universal program which thus contains all P_e^{E} and $P_{k(e)}^{\mathsf{E}}$. If any $P_e^{\mathsf{E}}(m,y)\!\downarrow$ then $P_{k(e)}^{\mathsf{E}}(m,y)\!\downarrow w_{\sigma_0}$ as above; from w_{σ_0} we may compute a code for $L_{\sigma_0}[y]$, run $P_e^{\mathsf{E}}(m,y)$ inside this model and see that definably over it we have course-of-computation code u as in the statement of the theorem.

NB: We have yet to show the next two even for $\mathsf{I} = \mathsf{E}$.

Lemma 3.10. $A \subseteq \omega$ *is* I*-semirecursive* $\Longleftrightarrow A \in \Sigma_1\big(L_{\alpha_0^{\mathsf{I}}}[\mathsf{I}]\big)$.

Lemma 3.11. $A \subseteq {}^{\omega}\omega$ *is* I*-semirecursive* $\Longleftrightarrow$ *there is some* Σ_1 φ *so that*
$$x \in A \longleftrightarrow\ L_{\alpha_0^{\mathsf{I},x}}[\mathsf{I}, x] \vDash \varphi[x, \mathsf{I}].$$

Exercise 3.3. Show that there is a p.r. function l so that if $\{e\}^{\mathsf{E}}(m,x)\!\downarrow$ and thus has a wellfounded computation tree $\mathfrak{T}^{\mathsf{E}}(e,m,x)$ then $\{l(e)\}^{\mathsf{E}}(m,x)$ computes a code for its tree rank.

3.2 Nested Extendible Pairs

Bound up with the notion of levels of a computation tree is that of the depth of nesting of ordinals which we proceed to analyse. This will be crucial in our investigation of the ordinal α_0^{E}.

Definition 3.12. *For* $m \geq 1$ *an* m-depth Σ_2-nesting, *or just* m-nesting, *of an ordinal* α *is a sequence* $(\zeta_n, \sigma_n)_{n<m} = (\zeta_0, \ldots, \zeta_{m-1}, \sigma_{m-1}, \ldots\sigma_0)$ *so that*
(i) if $m = 1$ *then* $\zeta_0 < \alpha < \sigma_0$;
(ii) if $0 < n+1 < m$ *then* $\zeta_n \leq \zeta_{n+1} < \alpha < \sigma_{n+1} < \sigma_n$;
(iii) if $k < m$ *then* $L_{\zeta_k} \prec_{\Sigma_2} L_{\sigma_k}$.

3.2.1 The Σ_2-extendibility tree

(This subsection is not needed for what follows.) There are a number of things to discover about nestings. One can define a tree of such pairs, as follows.

Definition 3.13. *(The Σ_2-extendibility tree) We let $(\mathcal{T}, \prec)$ be the natural tree on such pairs under inclusion: as follows: if (ζ', Σ'), $(\bar\zeta, \bar\Sigma)$ are any two countable Σ_2-extendible pairs, then set $(\bar\zeta, \bar\Sigma) \prec (\zeta', \Sigma')$ iff $\zeta' \leq \bar\zeta < \bar\Sigma < \Sigma'$.*

- If we had allowed the inequality $\bar\Sigma \leq \Sigma'$ rather than a strict inequalitiy in the last definition we could have defined a larger strict partial order $\prec'$, and a larger tree $(\mathcal{T}', \prec')$; however this would not have been wellfounded: if $L_\Sigma \vDash \Sigma_2$-Sep then it is easy to see that $(\mathcal{T}' \restriction \Sigma + 1, \prec')$ is illfounded.

Lemma 3.14. *Let δ be least such that $L_\delta \vDash \Sigma_2$-Sep. ; let α be maximal so that $(\mathcal{T}' \restriction \alpha, \prec')$ is wellfounded (where* $\mathrm{Field}(\mathcal{T}' \restriction \alpha) =_{\mathrm{df}} \{(\zeta, \Sigma)$ an extendible pair $\mid \Sigma < \alpha\}$*). Then $\delta = \alpha$.*

Proof: $(\leq)$: Suppose $\delta > \alpha$. Then $(\mathcal{T}' \restriction \delta, \prec')$ is illfounded. So there is an infinite sequence of extendible pairs (ζ_n, Σ_n) with $(\zeta_{n+1}, \Sigma_{n+1}) \subset (\zeta_n, \Sigma_n)$. By wellfoundedness of the ordinals there is some m_0 so that for all $n > m_0$ all Σ_n are equal to a fixed Σ, whilst $\zeta_n < \zeta_{n+1}$. Let $\zeta^* = \sup_n \zeta_n$. Then we have for $n > m_0$ $L_{\zeta_n} \prec_{\Sigma_2} L_{\zeta_{n+1}} \prec_{\Sigma_2} L_{\zeta^*}$. Then ζ^* is not Σ_2-projectible, and hence $L_{\zeta^*} \vDash \Sigma_2$-Sep. But $\zeta^* < \delta$. Contradiction.

$(\geq)$: $L_\delta \vDash \Sigma_2$-Sep. Then S^2_δ is unbounded in δ. Let $\delta_i < \delta_{i+1}$ be a cofinal sequence, for $i < \omega$. Then $\langle (\delta_i, \delta) \mid i < \omega \rangle$ is a $\prec'$-descending sequence in $\mathcal{T}' \restriction \delta + 1$. So $\alpha \leq \delta$. $\square$

3.2.2 Infinite depth nestings

We shall want to consider non-standard admissible models (M, E) of KP together with some other properties. We let $\mathrm{WFP}(M)$ be the wellfounded part of the model. By the so-called 'Truncation Lemma' it is well known (*v.* [1]) that this well founded part must also be an admissible set. Usually for us the model will also be a countable one of "$V = L$" (or $L[\boldsymbol{x}]$). Let M be such and let $\alpha = \mathrm{On} \cap \mathrm{WFP}(M)$. By the above α is thus an 'admissible ordinal', *i.e.* L_α will also be a KP model. An 'ω-depth' nesting cannot exist by the wellfoundedness of the ordinals. However an ill founded model M when viewed from the outside may have infinite descending chains of 'M-ordinals' in its illfounded part. These considerations motivate the following definition.

Definition 3.15. *An* infinite depth Σ_2-nesting of α based on M *is a sequence $(\zeta_n, s_n)_{n<\omega}$ with :*
(i) $\zeta_n \leq \zeta_{n+1} < \alpha \subset s_{n+1} \subset s_n$; (ii) $s_n \in \mathrm{On}^M$; (iii) $(L_{\zeta_n} \prec_{\Sigma_2} L_{s_n})^M$.

Thus the s_n form an infinite descending E-chain (where, as above, E is the membership relation of the illfounded model) through the illfounded part of the model M.

In order for there to exist a non-standard model with an infinite depth nesting (of the ordinal of its wellfounded part) then the wellfounded part will already be a relatively long countable initial segment of L (it is easy to see that if $\zeta = \sup_n \zeta_n$ then already $L_\zeta \vDash \Sigma_1$-Separation, hence there can be no infinite depth nesting of ω_1^{ck} for example). The next two exercises gives examples of such nested ordinals.

Exercise 3.4. Let δ be least so that ${}^\omega\omega \cap L_\delta$ is a model of Π^1_3-CA_0, or more generally δ countable with $L_\delta \vDash \Sigma_2$-Separation, and let (M, E) be any admissible non-wellfounded end extension of L_δ with L_δ as its wellfounded part. (Such (M, E) exist by the Barwise Compactness Theorem.) Then there is an infinite depth nesting of δ based on M. [Hint: If $\zeta_0 \in S^2_\delta =_{\mathrm{df}} \{\tau < \delta \mid L_\tau \prec_{\Sigma_2} L_\delta\}$, then ζ_0 has arbitrarily large Σ_2-end extensions L_τ for $\tau < \delta$ - namely any L_τ with $\tau \in S^2_\delta$. So by overspill considerations, L_{ζ_0} must have an ill-founded Σ_2-end extension $(L_{t_0})^M$. Repeat for $\zeta_1 > \zeta_0$ *etc.*]

Exercise 3.5. Let γ_0 be least such that there is $\gamma_1 > \gamma_0$ with $L_{\gamma_0} \prec_{\Sigma_2} L_{\gamma_1} \models$ KP. (i) Show that there is an infinite depth nesting of γ_1 based on some (or any) illfounded end extension M of L_{γ_1}. (ii) Show that $\gamma_1 \geq \beta_1$ where the latter is the least ordinal of the type defined in Ex.3.1. [Hint: For (i) Note that $T^2_{\gamma_0} = T^2_{\gamma_1}$, and moreover that every $x \in L_{\gamma_0}$ is Σ_2-definable in L_{γ_0} (by the leastness of γ_0). Further that the Σ_2-sentences of $\mathcal{L}_{\dot{\in}}$ are partioned into $S \cup U$,where the sentences in S are *stable*, that is have the same truth value in all L_α, $\alpha \in [\gamma_0, \gamma_1]$, and those in U are *unstable* and alternate unboundedly in both γ_0 and γ_1. Use KP to argue that there are unboundedly many $\gamma \in (\gamma_0, \gamma_1)$ with $T^2_\gamma = T^2_{\gamma_0}$. By leastness of γ_0 we have that $L_{\gamma_0} \prec_{\Sigma_2} L_\gamma$. Now apply the method of the last Exercise.]

Definition 3.16. *Let β_0 be the least ordinal β so that L_β has an admissible end-extension (M, E) based on which there exists an infinite depth Σ_2-nesting of β.*

Exercise 3.6. Show that for any infinite depth nesting of an ordinal α based on the model (M, E), that the E-infimum of the upper interval ends s_n in Def 3.15 must be α itself.

Exercise 3.7. Show that if there is an infinite depth nesting of some ordinal β_1 where the lower ordinals ζ_n of the surrounding nesting sequence are kept constant, then $\beta_1 > \beta_0$.[Hint: See "Lemma 16" of ⟨cite|AW⟩.]

Exercise 3.8. Give examples of ordinals γ, δ where there is an infinite nesting of γ where the lower ordinals ζ_n are kept constant, but no nesting of γ with the ordinals ζ_n strictly increasing; but for δ both kinds of nesting of δ are possible.

Exercise 3.9. Show that if γ has an infinite nesting based on (M, E) with the ordinals ζ_n strictly increasing with supremum $\lambda < \gamma$ then γ has an infinite nesting based on some (M', E'), with all lower ordinals $\zeta_n = \lambda$. Can $(M', E') = (M, E)$? [Hint: Show that $J_\lambda \prec_{\Sigma_2} J_\gamma \models$ KP and then we can use Ex.3.5. Suppose $J_\gamma \models \varphi(p)$ for some $p \in J_\lambda$. By underspill there are $\gamma' < \gamma$, $\zeta' < \lambda$ with $J_{\zeta'} \prec_{\Sigma_2} J_{\gamma'} \models \varphi(p)$. But $\zeta' \in S^1_\lambda$. So $J_\lambda \models \varphi(p)$.]

Question: Suppose α has an infinite nesting based in a model M, having intervals with a constant lower ordinal ξ, does it have another nesting based in some M' with increasing ζ_n which have supremum that ξ? (Converse to Ex. 3.9) (It is easy to see that we cannot have this with $M = M'$.)

(By Ex. 3.5 the answer is "No".)

3.3 The course of computations

We want to investigate the course of possible ittm computations. We shall consider just the functional E, and most of the time just consider computations of the form $\{e\}^{\mathsf{E}}(m)$ for simplicity's sake as the methods parameterize to reals uniformly. We've seen that a computation $\{e\}^{\mathsf{E}}(m)$ can, by making use of suitable queries, 'import' into its scratch tape the OT of any $\{f\}^{\mathsf{E}}(k, y)$ (when convergent) for any f, m, y for which it can formulate the question. It can also calculate a code for the order type of the upper end, σ, of the looping Σ_2-extendible pair (ξ, σ) which witness the final looping status of $\{f\}^{\mathsf{E}}(k, y)$. The latter indicates that the lengths of convergent E-computations are likely to be all E-recursive ordinals. This indeed will turn out to be the case. However that does not yield a characterisation of such ordinals or even a bound on those lengths. This we now want to investigate.

Recall that the snapshot at time $\alpha < H(\mathsf{E}, e, \boldsymbol{m}, \boldsymbol{x})$ is $s(\alpha) = \langle I(\alpha), R(\alpha), \langle C_i^{\nu(\alpha)}(\alpha) | \, i < \omega \rangle \rangle$. Here for $P_e^{\mathsf{E}}(m)$, the functional E is simplicity itself, hence the snapshot function $s \restriction \eta$ for limit $\tau > \eta$ is $\Delta_1^{J_\tau}$, whilst $s(\tau)$ itself is $\Sigma_2^{J_\tau}$ by use of the liminf rule. Consequently if (ξ, σ) is an extendible pair, then we shall have that $s(\xi) = s(\sigma)$, and in particular the computation $P_e^{\mathsf{E}}(m)$ is being carried on at the same node $\nu(\xi) = \nu(\sigma)$ at these two times. If this node is $\nu(0)$, the topmost node, then this is a pair of final looping snapshots. If this node is some other $\nu(\alpha)$ at some level $\Lambda = k > 0$, then this is a pair of final looping snapshots in some subcomputation $P_{e_\iota}^{\mathsf{E}}(m_\iota, y_\iota)$ - immediately after which at time $\sigma + 1$ some value, and control, is passed up to the node immediately above $\nu(\alpha)$ in the tree. Similar considerations are at play in the following.

Lemma 3.17. *Suppose (ξ, σ) is an extendible pair, and $P_e^{\mathsf{E}}(m)$ a computation. Then for all $\alpha \in (\xi, \sigma)$, $\Lambda(e, m, \alpha) \geq \Lambda(e, m, \xi) = \Lambda(e, m, \sigma)$.*

Proof: The latter equation follows by Σ_2-reflection and the liminf rule, as above, this will mean the snapshots at ξ and σ are identical. However if $\exists\alpha \in (\xi, \sigma)$, $\Lambda(e, m, \alpha) < \Lambda(e, m, \xi)$, then again by Σ_1-reflection, there are unboundedly many $\beta < \xi$ with $\Lambda(e, m, \alpha) = \Lambda(e, m, \beta)$. Again by Liminf applied to the levels at stage ξ, $\Lambda(e, m, \alpha) \geq \Lambda(e, m, \xi)$ - a contradiction. QED

Similarly:

Lemma 3.18. *(i) Suppose we have a 2-nesting $\zeta_0 < \zeta_1 < \Sigma_1 < \Sigma_0$. Suppose no $\alpha < \zeta_0$ of the overall computation of $P_e^{\mathsf{E}}(m)$ is the start of a final loop and $\Lambda(e, m, \zeta_0) = k$. Then no $\alpha < \zeta_1$ is the starting point of a final loop, and moreover $\Lambda(e, m, \zeta_1) \geq k + 1$.*
(ii) More generally if we have a p-nesting $\zeta_0 < \cdots < \zeta_{p-1} < \Sigma_{p-1} < \cdots < \Sigma_0$ and we suppose again that no $\alpha < \zeta_0$ is the start of a final loop in the computation of $P_e^{\mathsf{E}}(m)$, and that $\Lambda(e, m, \zeta_0) = k$. Then $\Lambda(e, m, \zeta_{p-1}) \geq k + p - 1$.

Proof: We consider first just a $p = 2$-nesting. By Σ_2-reflection and the liminf rule, as above, this will mean the snapshots at ζ_0 and Σ_0 are identical; hence $P_e^{\mathsf{E}}(m)$ is still running at depth k at Σ_0 at the same node $\nu(\zeta_0) = \nu(\Sigma_0)$. Suppose $k = 0$. Thus $P_e^{\mathsf{E}}(m)$ has as its first repeating loop $[\zeta_0, \Sigma_0]$. Suppose for a contradiction that control is at level 0 also at time ζ_1 (and again also at Σ_1). So again $P_e^{\mathsf{E}}(m)$ has looping snapshots at (ζ_1, Σ_1). However this is a Σ_1-fact about $P_e^{\mathsf{E}}(m)$ that L_{Σ_0} sees: "*There exists a 2-extendible pair $(\bar\zeta, \bar\Sigma)$ with $P_e^{\mathsf{E}}(m)$ having identical snaphots at level 0 at $(\bar\zeta, \bar\Sigma)$.*" But then there is such a pair $\bar\zeta < \bar\Sigma < \zeta_0$ and $P_e^{\mathsf{E}}(m)$'s computation is again looping at $\bar\zeta$ contrary to the supposition on ζ_0.

The argument for $k \geq 1$ is very similar: if $\mathrm{liminf}_{\alpha\to\zeta_0} \Lambda(e, m, \alpha) = \Lambda(e, m, \zeta_0) = k$, then $\mathrm{liminf}_{\alpha\to\Sigma_0} \Lambda(e, m, \alpha) = k$ also. Again, if it entered the interval (ζ_1, Σ_1) at this same level k it would loop there with identical snapshots at ζ_1, Σ_1, and by the same reflection argument applied repeatedly would do so not just once but unboundedly below ζ_0 at the same level k. But after each successful loop at level k, control passes up to level $k - 1$. However then $\mathrm{liminf}_{\alpha\to\zeta_0} \Lambda(e, m, \alpha) \leq k - 1$. Contradiction! QED

Corollary 3.19. *Suppose α is n-nested (for some $n \geq 1$). Then $P_e^{\mathsf{E}}(m)$ is not exhibiting final looping behaviour at or before α only if $\Lambda(e, m, \alpha) \geq n$.*

Proof: If $n = 1$, $\zeta_0 < \alpha < \Sigma_0$ with (ζ_0, Σ_0) an extendible pair, and were $\Lambda(e, m, \alpha) = 0$, then by Σ_2 reflection we should have $\Lambda(e, m, \beta) = 0$ for unboundedly many $\beta < \zeta_0$. But then $\Lambda(e, m, \zeta_0) = 0$ by the Liminf rule, and (ζ_0, Σ_0) is a final looping pair in the computation, and $P_e^{\mathsf{E}}(m)$ is exhibiting final looping behaviour at or before α. For $n > 1$ use induction. (Exercise) QED

Lemma 3.20. *(Boundedness Lemma for computations recursive in E) Let β_0 be the least infinitely nested ordinal in some ill-founded model M with $\mathrm{WFP}(M) = L_{\beta_0}$. Let $\bar\alpha_0$ be least with $L_{\bar\alpha_0} \prec_{\Sigma_1} L_{\beta_0}$. Then any computation $P_e^{\mathsf{E}}(m)$ which is not already convergent by time $\bar\alpha_0$ has an ill-founded tree $\mathfrak{T}^{\mathsf{E}}(e, m)$.*

Proof: Let $\zeta_0 < \cdots < \zeta_n < \cdots\ \beta_0 \cdots \subset s_n \subset \cdots \subset s_0$ witness the infinite nesting at β_0 in M. By the definition of $\bar\alpha_0$ there is no least $\alpha \in [\bar\alpha_0, \beta_0)$ so that L_α sees that $P_e^{\mathsf{E}}(m)$ has a repeating looping snapshot as this would be a Σ_1-fact true in L_{β_0}; but then by Σ_1-reflection, it is true in $L_{\bar\alpha_0}$ and $P_e^{\mathsf{E}}(m)$ would then be convergent before $\bar\alpha_0$. However if $P_e^{\mathsf{E}}(m)$ has not failed before β_0, it will do so by β_0: the previous lemma shows that $\Lambda(e, m, \zeta_n) < \Lambda(e, m, \zeta_{n+1})$ holds in M. But these level facts are absolute to V, as they are grounded just on the part of the computation tree being built in L_{β_0} as time goes towards β_0; so $P_e^{\mathsf{E}}(m)$'s computation tree $\mathfrak{T}(e, m)$ will have an illfounded branch at time β_0. QED

The above then shows that the initial segment $L_{\bar{\alpha}_0}$ of the L-hierarchy contains all the information concerning looping or convergence of computations of the form $P_e^{\mathsf{E}}(m)$. A computation may then continue through the wellfounded part of the computation tree for the times $\beta < \beta_0$ but if so, it will be divergent. Relativisations to real inputs $\boldsymbol{x}$ are then straightforward by defining $\beta_0(\boldsymbol{x})$ as the least such that there is an infinite nesting based at that ordinal in the $L[\boldsymbol{x}]$ hierarchy *etc.*

[XXX Is the next exercise correct?? There might be another universal comp. $P_{e'}^{\mathsf{E}}(m)$ with a different arrangement of descent? XXX]

Exercise 3.10. Let β_0 be the least ordinal that supports an ω-nesting. Show that there is a maximal sequence $\zeta_i < \zeta_{i+1} < \beta_0$ for $i < \omega$, which are the bottom ends of nesting Σ_2-pairs in *any* ω-nesting sequence based on β_0. [Hint: The descending sequence of levels Λ in $P_e^{\mathsf{E}}(m)$ occur at certain ordinal times independent of (N, E).]

Having ascertained $\bar{\alpha}_0$ as an upper bound for convergent computations of the form $\{e\}^{\mathsf{E}}(m)$, we now look to show that this is best possible. We have seen that we can compute codes for ordinals for increasing levels in the E^α-hierarchy, indeed computation of the next element of E^k can be effected by a computation using a tree $\mathfrak{T}^{\mathsf{E}}$ of rank k. We also have the natural finite depth of nesting of convergent computations. We need to have computations that approach β_0 in length. A convergent computation that somehow seemingly required infinite depth nesting would seem to be impossible. This suggests the following formulation.

We attempt to define a function t with domain ω by a Π_1-recursion, through finite approximation functions defined on initial segments of ω. Thus to some function t, but *via* approximations $t \restriction k+1$ where, for $i < k$, $\{t_{i+1}\}$ is Π_1 definable in $\langle t_0, ..., t_i\rangle$. We shall implement the process as defining a candidate for t_k as being Π_1-definable over some L_ξ where ξ is extendible. The process will calculate larger and larger extendibles ξ' and check that t_k, and indeed all the previous t_i for $i < k$ still satisfy the same Π_1 clauses over $L_{\xi'}$ as for the earlier L_ξ. If so then an attempt to define a non-empty $\{t_{k+1}\}$ is made over $L_{\xi'}$. On the other hand if for some least i_0, t_{i_0} fails to satisfy the Π_1 clause of its definition over $L_{\xi'}$, then all the current approximating values t_j for $i_0 \leq j \leq k$ are abandoned, and the process continues with a search for a new approximating value $\{t'_{i_0}\}$ (*if it exists*), and advances to the next extendible level $L_{\xi''}$. We emphasise existence here, since *prima facie* there is no guarantee at any point of there being a non-trivial candidate in $L_{\xi'}$ for t'_{i_0}, and one may simply have to move to a higher $L_{\xi''}$ to look again.

As just stated then, there is, in general, no guarantee that any instances of a function t being defined in this quasi-recursive fashion will eventually be definable at some level L_ξ. Nor should we expect that approximations will grow in a monotone fashion: Π_1 properties fail to be upwards persistent.

Nevertheless we shall be able to find a Π_1-definable process which will have the following essential property: suppose $\zeta_0 < \zeta_1 < \alpha < \sigma_1 < \sigma_0$ is a nesting of α as above and $t \restriction 1 = t_0 \in L_{\zeta_0}$ results from applying our Π_1 definition over L_{ζ_0}. Then there is to be guaranteed a candidate $t_1 \in L_{\zeta_1}$ to extend $t \restriction 1$ to $t \restriction 2$. In fact we shall have slightly more than that. Let $\varphi(v_0, v_1)$ be our intended Π_1-formula for defining approximations on ω to such a t, and the L-least x such that $\varphi(x, t \restriction k)^{L_\xi}$ is to be set as $t(k)$ to extend the approximation to $t \restriction k+1$. Let $\tilde{\varphi}(u)^{L_\xi}$ hold if u is a finite sequence and for all $k < \mathrm{dom}(u)$ u_k is L-least s.t. $\varphi(u_k, u \restriction k)^{L_\xi}$.

$\Upsilon(\varphi)$: Suppose $\zeta_0 < \zeta_1 < \cdots < \zeta_k < \alpha < \sigma_k < \cdots < \sigma_1 < \sigma_0$ is any $k+1$-nesting (for variable k); then

a) there exists $t \restriction k \in L_{\zeta_{k-1}}$ with $\tilde{\varphi}(t \restriction k)^{L_{\zeta_{k-1}}}$ (for $k \geq 1$);

b) there exists $t_k \in L_{\zeta_k}$ to extend $t \restriction k$ to a $t \restriction k+1$, thus with $\tilde{\varphi}(t \restriction k+1)^{L_{\zeta_k}}$;

c) if the $t \restriction k$ from a) satisfies also $\tilde{\varphi}(t \restriction k)^{L_{\sigma_{k-1}^{++}}}$, then a t_k satisfying b) can be found in $L_{\zeta_{k-1}}$.

Note that by Σ_2-reflection, we have, concerning part b) that $\tilde{\varphi}(t \restriction k)^{L_{\zeta_{k-1}}} \longleftrightarrow \tilde{\varphi}(t \restriction k)^{L_{\sigma_{k-1}}}$ in any case. Clause c) says that if $t \restriction k$ survives satisfying $\tilde{\varphi}$ a bit further in to the next two admissibles $L_{\sigma_{k-1}^{++}}$, then we already have a suitable $t \restriction k$ earlier in $L_{\zeta_{k-1}}$. Clearly the property implies that b) holds for any k' with $0 \leq k' \leq k$. In any case, for any $p+1$-nesting we can define a candidate function $t \restriction p+1$ over L_{ζ_p}. All that would be needed to define a complete function t at L_α is an ω-nesting! We have yet to justify that there is any φ so that $\Upsilon(\varphi)$ holds. We simply assert at this point that it exists:

Lemma 3.21. *There is a Π_1 φ so that $\Upsilon(\varphi)$ holds.*

3.4 Instantiating Υ

The first proof concerning β_0 as the least upper bound for wellfoundedness of computation trees of ittm-recursions in E (on integer input) went *via* a proof Σ^0_3-Determinacy. The principle Υ was validated for a $\tilde{\varphi}$ that looked for successive non-losing quasi-strategies for games $G(B_n)$ where a Σ^0_3 set $A = \bigcup_n B_n$ for $B_n \in \Pi^0_2$, which Player *II* is to win. It was known that an overall strategy for Player *II* in $G(A)$ was to be found definably over L_{β_0} (*cf.* [17]). Hence having a generalized ittm recursion that sought for such quasi-strategies brought one ever closer to this level of the L-hierarchy.

Hachtman in [4] then showed that the reals of any admissible model $L_\beta \vDash V = \mathrm{HC}$ on which could be based an ω-nesting, formed a model of Π^1_2-monotone induction. He also showed (*inter alia*) that L_{β_0} yielded the least β-model of Π^1_2-MI. That is, he showed that the reals of the least level of L that satisfied Π^1_2-MI, were the reals of L_{β_0}. However this also went indirectly *via* the result on Σ^0_3-Determinacy first holding definably over L_{β_0}.

It was thus desirable to have an ittm recursive in E procedure which was more natural in the context of such recursions in order to, *e.g.*, characterise the halting problem H for such recursions/machines, rather than searching for quasi-strategies in a game theoretic fashion.

What we effect below is a search for admissible levels of L for finitely many ordinals, where it is consistent for those ordinals to be the left hand end of a nested sequence in an ω-model which is an end extension of that level. This sequence $(t_0, \ldots, t_k)$ will be of approximations to an eventual ω-sequence witnessing an infinite depth nesting in some ω-model which as we have seen yields an upper bound for the ranks of wellfounded computation trees of ittm-recursions in E (on integer input) - see the Boundedness lemma above Lemma 3.20.

We use the notion of an infinitary logic: for α an admissible, we set $\mathcal{L}_\alpha =_{\mathrm{df}} L_{\omega_1,\omega} \cap L_\alpha$ and is thus an 'admissible fragment' in the sense of [1], [7]. More specifically we let Γ_α be the axioms of $\mathcal{L}_\alpha$ comprising of: (i) KP; (ii) the $\mathcal{L}_\alpha$-infinitary atomic diagram of L_α; (iii) for any $y \in L_\alpha$: $\forall x \big(x \in c_y \longrightarrow \bigvee_{u \in y} x = c_u \big)$ (we assume there are constants c_u in $\mathcal{L}_\alpha$ for each $u \in L_\alpha$). Any model of Γ_α has then (a transitive copy of) $(L_\alpha, \in)$ contained in its wellfounded part (by (ii)) and is indeed an end-extension of $(L_\alpha, \in)$ by (iii). We set, with $\mathrm{WFD}(A)$ denoting the wellfounded part of a structure A:

$\Phi(\alpha, (t_0, \ldots, t_k)) \equiv$
"It is consistent in $\mathcal{L}_\alpha$-logic that there exists an end-extension $(N, E) \supset (L_\alpha, \in)$
$[\mathrm{WFD}(N) \supseteq L_\alpha \wedge \exists s_k, \ldots, s_0 \in \mathrm{On}^N((t_0, \ldots, t_k, s_k, \ldots, s_0)$ *forms a $k+1$-nesting with $t_k < \alpha$)]."*

Then $\Phi(\alpha, (t_0, \ldots, t_k))$ is a $\Pi^{L_\alpha}_1$-expressible assertion about the sequence $(t_0, \ldots, t_k)$ (since a *consistency property,* containing the theory Γ_α together with the assertion about the existence of a $k+1$ nesting *etc.*, if it exists, is so definable over L_α, and thus a model (N, E) of the Γ_α axioms with the nesting property satisfying the property Φ exists in L_{α^+}, the next admissible set.)

Lemma 3.22. *If (ξ, σ) is an extendible pair, there is $t_0 < \xi$ so that*

$$L_\sigma \vDash \text{“}\exists t_0 \forall \alpha \in (t_0, \sigma)(\alpha \in \mathrm{ADM} \longrightarrow \Phi(\alpha, (t_0)))\text{”}.$$

Proof: Note first that $\forall \alpha \in (\xi, \sigma)(\alpha \in \mathrm{ADM} \longrightarrow \Phi(\alpha, (\xi)))$. This is because the $\mathcal{L}_\alpha$-theory of L_α is consistent with there being such an extension in which α is 1-nested, since $(N, E) = (L_\sigma, \in)$ is such. Thus $L_\sigma \vDash$ “$\exists t_0 \forall \alpha \in (t_0, \sigma)(\alpha \in \mathrm{ADM} \longrightarrow \Phi(\alpha, (t_0)))$” (namely take $t_0 = \xi$). This is a Σ_2 statement and so by Σ_2 reflection goes down to L_ξ; there is such a $t_0 < \xi$, which ‘survives’ playing this role, meaning that it satisfies $\Phi(\alpha, (t_0))$ for all admissibles $\alpha < \sigma$ all the way through to σ. The moral is that as admissibles $\alpha \longrightarrow \xi$ then such a t_0 can be found in L_α, and eventually it will settle down on some value t which will be good up till σ. QED

Remark 3.23. Note however in the above if (ξ, σ) was the minimal extendible pair (ζ, Σ), that if $\alpha \in \mathrm{ADM}$ is least greater than Σ then a $t < \zeta$ cannot satisfy $\Phi(\alpha, (t))$: if there were some model as specified with $(N, E) \supset (L_\alpha, \in)$ we should have a 2-nesting in N, with $t < \zeta < \Sigma < s$. By Σ_1-reflection in N there would have to be an extendible pair (ζ', Σ') in L_t which is impossible. Thus such a t does not survive (playing its role) beyond Σ. On the other hand for other extendible pairs $(\bar{\xi}, \bar{\sigma})$, for example those that are themselves contained within another extendible pair (ξ, σ), then such a t can and does survive well past $\bar{\sigma}$; indeed it may do so till σ.

Remark 3.24. From Φ one can write down a Π_1 φ so that $\Upsilon(\varphi)$ and so that Lemma 3.21 holds. Namely, let $\tilde{\varphi}(t \restriction n+1)$ (or $\varphi(t_n, t \restriction n)$) be $\forall \alpha > t_n(\alpha \in \mathrm{ADM} \longrightarrow \Phi(\alpha, (t_0, ..., t_{n-1}, t_n)))$.

Lemma 3.25. $\Upsilon(\varphi)$ *holds.*

Proof: We verify $\Upsilon(\varphi)$a) and b) for $k = n$. Suppose α approaches from below ${}^n\zeta$, where $({}^n\zeta, {}^n\Sigma)$ is some n-extendible pair with $n \geq 1$ (it suffices to think of $n = 1$). Fix such a pair $({}^1\zeta, {}^1\Sigma)$. By Lemma 3.22 there is some $t_0 < {}^1\zeta$ which is good up to ${}^1\Sigma$ (just replacing ξ, σ there by the two ordinals ${}^1\zeta, {}^1\Sigma$ here). This verifies $\Upsilon(\varphi)$a) [k=1]. Recall that both by definition and Σ_2-reflection ${}^1\zeta$ and ${}^1\Sigma$ are limits of 1-extendible pairs $({}^0\zeta, {}^0\Sigma)$. Now let $({}^0\zeta, {}^0\Sigma) \subset ({}^1\zeta, {}^1\Sigma)$. We seek a candidate t_1 for an approximation (t_0, t_1). Let $\beta \in \mathrm{ADM} \cap ({}^0\Sigma, {}^1\Sigma)$. As $\Phi(\beta, t_0)$ holds in $L_{{}^1\Sigma}$ there is an end-extension $(N, E) \supset L_\beta$ so that in (N, E) the following hold:

(i) $L_{t_0} \prec_{\Sigma_2} L_{s_0}$ *for some* s_0; (ii) *there is some* t_1 *so that for admissible* $\alpha \in ({}^0\zeta, {}^0\Sigma)$ *it is consistent in* $\mathcal{L}_\alpha$*-logic that there is an end-extension* (N', E') *of* L_α, *and with* $s_1' < s_0'$ *in* $\mathrm{On}^{N'}$, *with* (t_0, t_1, s_1', s_0') *a 2-nesting.*

(Again, for part (ii), this is consistent because it is so in (N, E) as witnessed by $t_1 = {}^0\zeta$, $s_1' = {}^0\Sigma \wedge s_0' = s_0$.) Arguing just as in Lemma 3.22 $L_{{}^0\Sigma}$ sees this consistency about some t_1 in $\mathcal{L}_\alpha$-logic as stated in (ii), for $\alpha < {}^0\Sigma$, and by Σ_2-reflection, there is a (least) $t_1 \in (t_0, {}^0\zeta)$ for which this holds in $L_{{}^0\Sigma}$. This verifies $\Upsilon(\varphi)$b) [k=1].

So for a fixed t_0, for each $({}^0\zeta, {}^0\Sigma) \subset ({}^1\zeta, {}^1\Sigma)$ we can associate a t_1 for which “$\forall \alpha \in \mathrm{ADM}\Phi(\alpha, (t_0, t_1))$” holds in $L_{{}^0\Sigma}$. Note that we may keep t_0 fixed, but different intervals $({}^0\zeta, {}^0\Sigma) \subset ({}^1\zeta, {}^1\Sigma)$ may require different t_1, because a t_1 defined for an interval $({}^0\zeta, {}^0\Sigma)$ may not ‘survive’ beyond ${}^0\Sigma$. As $\alpha \longrightarrow {}^n\zeta$ by repeating the argument, for each nested $({}^1\zeta, {}^1\Sigma) \subset \cdots \subset ({}^n\zeta, {}^n\Sigma)$ there is some $(t_0, ..., t_{n-1})$, so that for each interval $({}^0\zeta, {}^0\Sigma)$ contained in $({}^1\zeta, {}^1\Sigma)$ we can associate a t_n for which “$\forall \alpha \in \mathrm{ADM}\Phi(\alpha, (t_0, t_1, ..., t_n))$” holds in $L_{{}^0\Sigma}$. Thus $\tilde{\varphi}(t \restriction n+1)$ holds in $L_{{}^0\Sigma}$.

This leaves $\Upsilon(\varphi)$c). We argue this for $k = 1$ and for larger k this is no different. Suppose that $t_0 < \zeta_1$ is such that $\tilde{\varphi}((t_0))^{L_{{}^{\sigma_1}++}}$. That is, for admissible $\alpha \leq \sigma_1^+$, there is an ω-model $(N, E) \supset (L_\alpha, \in)$ with some $s_0 \in \mathrm{ON}^N$ such that $((t_0, s_0)$ *is an extendible pair*$)^N$. Apply this with $\alpha = \sigma_1^+$ and for the resulting (N, E) we have $(\zeta_1, \sigma_1) \subset (t_0, s_0)$. Apply again the previous arguments inside (N, E): $(*)$ holds in N where:

$(*)$ “*There exists* t_1 *so that for any* $\beta \in (\zeta_1, \sigma_1)$ *it is* $\mathcal{L}_\beta$*-consistent that there exists an end-extension* (N', E') *of* L_β, *and with* $s_1', s_0' \in \mathrm{On}^{N'}$ *so that* (t_0, t_1, s_1', s_0') *is a nesting with* $t_1 < \beta \subset s_1'$”.

We note that it is indeed consistent with the atomic diagram of L_β, as this is witnessed in (N, E) by $(t_0, t_1 = \zeta_1, s_1' = \sigma_1, s_0' = s_0)$. By the Σ_2-reflection of L_{σ_1} down to L_{ζ_1} we have that there is such a $t_1 < \zeta_1$ as required for $\Upsilon(\varphi)$c). QED

4 A Least Upper Bound to Computation Lengths

We thus take the Π_1 formula φ from the last section, with the property $\Upsilon(\varphi)$ above, and with $\tilde{\varphi}(u) \equiv \forall k < \mathrm{dom}(u)(u_k$ is L-least so that $\varphi(u_k, u \restriction k))$ the associated Δ_2-formula. We shall then see how to define a recursion in E that fails, that is, has an illfounded computation tree, but it only becomes illfounded at stage β_0. It thus outlasts all the convergent computations $\{e\}^{\mathsf{E}}(m)$, which by the Boundedness Lemma 3.20 all converge before $\bar{\alpha}_0 < \beta_0$. One might be curious about the interval $(\bar{\alpha}_0, \beta_0)$. At risk of whimsy (something always to be avoided) - this appears as some kind of event horizon: any computation $\{e\}^{\mathsf{E}}(m)$ that enters this region is destined to fail with an illfounded computation tree; the computation continues, remaining wellfounded for all $\alpha < \beta_0$, but eventually falls into the black hole of β_0.

We give a scheme for a process that will be implementable as a recursion in E. In order for its description to be manageable we shall use some abbreviations. To recapitulate, we have seen above that recursions in a suitable K which for queries $Q^{\mathsf{K}}(e,\ m,\ y)$ formally return $\mathsf{K}(z)$ when $\{e\}^{\mathsf{K}}(m, y)\downarrow z$, can usefully be organised as sequences of such queries which have (a) the effect of returning simply a check as to whether $\{e\}^{\mathsf{K}}(m, y)\downarrow\backslash\uparrow$, or indeed (b) the infinite sequence of values $z(0), \cdots$ if $\{e\}^{\mathsf{K}}(m, y)\downarrow z$.

We thus can ask for the whole result $z \in {}^{\omega}\omega$ where $\{e\}^{\mathsf{K}}(m,\ y)\downarrow z$ during a computation call. We can also ask for the *length* σ of the looping computation $\{e\}^{\mathsf{K}}(m,\ y)\downarrow z$ or its overall length $H(\mathsf{E}, e, m,\ y)$ (see Lemma 3.9), and get either in the form of a code $W_\sigma \in \mathrm{WO}$; we can also ask for $L_\sigma[z]$ *etc.*, or for the next admissible ordinal σ^+ *etc.* We shall just describe "general queries" as those of this form that can be realised by (perhaps infinite) sequences of official queries. We regard such general queries as passing infinite amounts of information to lower levels of the computation tree as data for that subcomputation; and for the results, again as infinite sequences or amounts of information to be passed up to the controlling program at the next level up. The queries we shall freely formulate in English to describe a process to be transacted at the next level down without involving us in all of the formal *minutiae*. We shall do this without further comment in the confidence that the enthusiastic reader (if there are any) could if they wished, with effort, convert these general queries into the officialese of our formalism.

We now outline the algorithm at the various levels of computation in the oracle calls of a (master) computation which runs at node ν_0 (the only node at level $\Lambda = 0$). We proceed by describing the actions of the programmes being called, which the reader, as we have just said, may reformulate as official queries to the E-functional as oracle.

At the end of the description we justify the claim that this is a bona fide E-recursion of the form $\{e_0\}^{\mathsf{E}}(k, t, W)$. In the sequel W is intended to be a subset of ω coding a well ordering; if the well ordering is of length δ then it is intended to be the L-least well ordering of this length, and we indicate this by writing W_δ. (In the region of L under discussion (i) $L_\delta \vDash$ "Every set is countable"; (ii) there is always a $\Sigma_2^{L_\delta}$ map (possibly requiring a parameter) of ω onto L_δ (or a $\Sigma_3^{L_\delta}$ such map parameter free); (iii) uniformly definable Σ_n-skolem functions for the J_δ (or L_δ if $\mathrm{Lim}(\delta)$); W_δ is thus always an element of $L_{\delta+1}$.) During the proof ${}^0M$ is a variable for a structure $(L_\sigma, \in)$ with some $\zeta < \sigma$ so that (ζ, σ) is an 1-extendible pair.

The programme $\{e_0\}^{\mathsf{E}}(\langle 0, t_0, W_\sigma\rangle)$ operates as follows, starting with $t_0 = \varnothing$ and $W_\sigma = W_\omega$:

At ν_0 (so at $\Lambda = 0$):

The master computation $\{e_0\}^{\mathsf{E}}(0)$ is run at this level, and computes successive 1-extendible structures of the form ${}^1M$ of increasing length, looking for a candidate t_0 for which $\varphi(t_0)$ which it may write to its OT. (Whilst doing this it writes out the increasing theories T^2_α for successive α to a register $\mathcal{R}_0$ for book-keeping purposes.) When it finds a t_0 in some structure ${}^0M$ so that $(\varphi(t_0))^{{}^0M}$, (which actually it does immediately by our $\Upsilon(\varphi)$a) for $k = 1$) it launches a query to a subroutine one level down which essentially asks ?Q: *Is there a stable candidate for t_1 so that $\tilde{\varphi}(\langle t_0, t_1\rangle)$?* Slightly more formally:

Q^1: *Does $\{e_0\}^{\mathsf{E}}(\langle 1, \langle t_0\rangle, \sigma\rangle)$ find a stable candidate for t_1 using t_0 so that $\tilde{\varphi}(\langle t_0, t_1\rangle)$ and starting from the ordinal $\sigma = \mathrm{On} \cap {}^0M$?*

Thus the information $x = \langle 1, \langle t_0\rangle, \sigma\rangle$ passed down in the query contains the 'current attempt' at t_0 and the 'current ordinal', that is the height of the 'current extendible structure' ${}^0M$. A 'stable candidate' is one which eventually settles down to a fixed value in successively longer extendible structures ${}^0M$, and could be written to the local OT.

At $\Lambda = 1$:

$P_{e_0}(\langle 1, \langle t_0\rangle, \sigma\rangle)$ computes further successive extendible structures of the form ${}^0M$ starting from σ, again writing theories T^2_α to a register $\mathcal{R}_1$. Various alternatives may happen:

• (1) *En route* $\varphi(t_0)$ may become invalid in some extendible structure (as it is a Π_1 property, and thus t_0 failed to survive and 'stably' satisfy φ in some longer structure). In which case, if this happens in some least such structure, and a new candidate t_0' is present, it and the current ordinal σ' (the ordinal hight of the current ${}^0M$) is passed back up to ν_0 at Λ_0. (If there is no t_0' yet then the 'empty candidate' $\varnothing$ is passed up instead.)

• (2) However if $\varphi(t_0)$ continues to hold in successive extendible structures, it may be that no potential t_1 is found. This means the process at this level continues until it loops. The answer to Q^1 is then "No". Setting the new σ' to be the length of this loop, then "No" and t_0, and $W_{\sigma'}$ are passed back up to $\Lambda = 0$, *i.e.* the node ν_0. (Strictly speaking, the looping length ordinal σ' is calculated *at* the higher level node, thus here at ν_0, rather than being 'passed up' to the node. But we say this in interest of brevity: we shall use this circumlocution without further comment below.)

• (3) Lastly if a t_1 is found so that $\tilde{\varphi}(\langle t_0, t_1\rangle)$ holds in some extendible structure ${}^0M$ of height σ', then a subcomputation query Q^2 is passed down to a new node at level $\Lambda = 2$:

Q^2: *Does $\{e_0\}^{\mathsf{E}}(\langle 2, \langle t_0, t_1\rangle, W_{\sigma'}\rangle)$ find a stable candidate for t_2 starting from $\langle t_0, t_1\rangle$ so that $\tilde{\varphi}(\langle t_0, t_1, t_2\rangle)$ and the ordinal σ'*?

At $\Lambda = 2$:

$P_{e_0}(\langle 2, \langle t_0, t_1\rangle, W_{\sigma'}\rangle)$ computes further successive extendible structures starting from σ', again writing theories T^2_α to a register $\mathcal{R}_2$. Various alternatives again may happen:

• (1) *En route* $\varphi(t_{i_0})$ may become invalid in some extendible structure for a least $i_0 < 2$. In which case the computation at this level HALTS, and if a new candidate t'_{i_0} is present, it and the current ordinal σ' is passed back up to $\Lambda = 1$, all other t_j for $j > i_0$ (if any) being discarded. (If there is no t'_{i_0} yet then the 'empty candidate' $\varnothing$ is passed up instead.) If $i_0 = 0$, then it is arranged that this information then is passed further up to level $\Lambda = 0$. Otherwise control is passed up to $\Lambda = 1$ where it now remains.

• (2) However if $\tilde{\varphi}(\langle t_0, t_1\rangle)$ continues to hold in successive extendible structures, it may be that no potential t_2 is found. This means the process at this level continues until it loops. The answer to Q^2 is then "No". Setting the new σ' to be the length of this loop, then "No" and $\langle t_0, t_1\rangle$, and $W_{\sigma'}$ are passed back up to $\Lambda = 1$.

• (3) Lastly if a t_2 is found so that $\tilde{\varphi}(\langle t_0, t_1, t_2\rangle)$ holds in some extendible structure ${}^0M$ of height σ', then on the first such occasion a query Q^3 is passed down to a new node at level $\Lambda = 3$:

Q^3: *Does* $\{e_0\}^{\mathsf{E}}(\langle 3, \langle t_0, t_1, t_2\rangle, W_{\sigma'}\rangle)$ *find a stable candidate for* t_3 *starting from* $\langle t_0, t_1, t_2\rangle$ *so that so that* $\tilde{\varphi}(\langle t_0, t_1, t_2, t_3\rangle)$ *and the ordinal* σ'?

And so forth. We trust the reader can see the successive formulations of Q^k *etc.* and the actions of $P^{\mathsf{E}}_{e_0}$ so intended.

One property of the construction is immediate:

Lemma 4.1. *The above construction has that for limit* α, $\Lambda(e_0, \alpha) = k > 0$ *implies that in an interval* (δ, α) $t \restriction k$ *is stable.*

Proof: For otherwise, unboundedly in α control would have been passed up to a level $\Lambda \leq k - 1$, and then by the liminf rule $\Lambda(e_0, \alpha) < k$. QED

Note 4.2. We also analysed the case that a loop occurred in which no appropriate extension of the approximation was found going from one ${}^0M$-structure to the next and the response to the query was negative. The alternative was phrased as (*cf.* the final case (3) when $\Lambda = 1 = k$ above) as t_k being defined in a structure. One might have asked: 'Where is the analysis that a loop occurred at level k in which all of $t \restriction k$ was stable throughout, and thus the answer to Q^k would have been "Yes", (as always at the end of a loop, followed by control passing up to $\Lambda = k - 1$)?' The following lemma shows that this behaviour does not occur.

Lemma 4.3. *For* $0 < k < \omega$, *if* $t \restriction k$ *is stably defined up to* $\bar{\Sigma}$ *(meaning* $\tilde{\varphi}(t \restriction k)^{L_\tau}$ *for all sufficiently large* $\tau < \bar{\Sigma}$*), and* $(\bar{\zeta}, \bar{\Sigma})$ *is a loop at* $\Lambda = k$ *resulting from a call from* $\Lambda = k - 1$ *to answer the query* Q^k *?* Does $\{e_0\}^{\mathsf{E}}(\langle k, t \restriction k, W_{\sigma'}\rangle)$ find a stable candidate for t_k starting from $t \restriction k$ and the ordinal σ'?, *then the response to* Q^k *is "No".*

Proof: The short answer is simply that if the response were "Yes", with t_k a stable candidate, and so that $\tilde{\varphi}(t \restriction k^\frown(t_k))^{L_\tau}$ held for all sufficiently large $\tau < \bar{\Sigma}$, then for all sufficiently large $\tau < \bar{\Sigma}$ control would be at a level $\Lambda(\tau) = m > k$, looking for some extension of $t \restriction k^\frown(t_k)$, and thus $(\bar{\zeta}, \bar{\Sigma})$ is a loop at some level $m \neq k$, contrary to hypothesis. □

Note 4.4. As the program runs there will eventually be subcomputation calls to arbitrary levels, as it uses approximations for as long as they survive fulfilling their role. But only after $\bar{\alpha}_0$ stages shall we be certain that t_0 really does stabilize to its final value. Thereafter we shall always have $\Lambda(e_0, (0, \varnothing, W_\omega), \alpha) \geq 1$. But only at β_0 shall we first have $\mathrm{Liminf}_{\alpha \to \beta_0} \Lambda(e_0, (0, \varnothing, W_\omega), \alpha) = \omega$ and so an illfounded computation.

We make some further observations on the flow of control during the recursion.

Lemma 4.5. *Let* $k \geq 1$. *Let* $({}^k\zeta, {}^k\Sigma) < \beta_0$ *be a* k*-extendible pair. At times during* $({}^k\zeta, {}^k\Sigma)$ *control will pass to depth at least* $\Lambda = k + 1$, *with queries* Q^{k+1} *asking that* $t \restriction k + 1$ *be stable and seeking a candidate for* t_{k+1}. *If additionally* $({}^k\zeta, {}^k\Sigma)$ *is itself nested inside some* $({}^{k+1}\zeta, {}^{k+1}\Sigma) \supset ({}^k\zeta, {}^k\Sigma)$, *then* $\Lambda(e_0, \alpha) \geq k + 1$ *for all sufficiently large* $\alpha < {}^k\Sigma$: $\exists \bar{\alpha} < {}^k\zeta \forall \alpha \in (\bar{\alpha}, {}^k\Sigma) \Lambda(e_0, \alpha) \geq k + 1$.

Proof: Formally a proof by induction on k, but really this is just a statement on the template of the construction. We discuss the case of $k = 1$; the larger values of k (or the variants with $t \restriction l$ stabilizing), just generalise this reasoning, so we leave this to the reader.

As $\alpha \longrightarrow {}^1\zeta$, by $\Upsilon(\varphi)$b) there is a $t_0 \in L_{{}^1\zeta}$ satisfying the Π_1 condition $(\varphi(t_0))^{L_{{}^1\zeta}}$.

We suppose that $({}^1\zeta, {}^1\Sigma) \supset ({}^0\zeta, {}^0\Sigma)$.

As $L_{{}^1\zeta} \prec_{\Sigma_1} L_{{}^0\zeta}$, $(\varphi(t_0))^{L_{{}^0\zeta}}$ also. But $\Upsilon(\varphi)$b)$[k=1]$ implies that there is $t_1 \in L_{{}^0\zeta}$ such that $\tilde{\varphi}(t_0 \frown t_1)^{L_{{}^0\zeta}}$. By our description of $P^{\mathsf{E}}_{e_0}$ then a query Q to $\Lambda = 2$ is launched asking if $t_0 \frown t_1$ is stable and seeking a t_2. (And actually by Σ_2-reflection such a query or queries are being acted out at level $\Lambda = 2$ unboundedly in ${}^1\zeta$ also.[CHECK XXXX?])

If also $({}^1\zeta, {}^1\Sigma)$ is nested inside $({}^2\zeta, {}^2\Sigma) \supset ({}^1\zeta, {}^1\Sigma)$, then $\Lambda(e_0, \alpha) \geq 2$ for all sufficiently large $\alpha < {}^1\Sigma$: $\exists \bar{\alpha} < {}^1\zeta \forall \alpha \in (\bar{\alpha}, {}^1\Sigma) \Lambda(e_0, \alpha) \geq 2$. This is because by $\Upsilon(\varphi)$ we now have a $t_0 < {}^2\zeta$ and a $t_1 < {}^1\zeta$ with $\tilde{\varphi}(t_0 \frown t_1)^{L_{{}^1\zeta}}$ and so $\tilde{\varphi}(t_0 \frown t_1)^{L_{{}^1\Sigma}}$. But this means the computation stays down at level greater than or equal to 2 for $\alpha \in [{}^1\zeta, {}^1\Sigma)$ as both t_0, t_1 are stable to $L_{{}^1\Sigma}$, and a t_2 is being sought. QED

Corollary 4.6. *If $\zeta_0 < \zeta_1 < \cdots < \zeta_k < \alpha < \sigma_k < \cdots < \sigma_1 < \sigma_0 < \beta_0$ is any $k+1$-nesting then for some $\delta < \zeta_0$, $\Lambda(e_0, \gamma) \geq k$ for $\gamma \in (\delta, \sigma_0)$.*

Lemma 4.7. *Control passes at a stage from $\Lambda = 1$ to $\Lambda = 0$ only when t_0 becomes unstable; in particular when either (a) calculating at $\Lambda = 1$ sees that t_0 is directly different from one ${}^0M_1$ to the next ${}^0M_2$; or (b) information has just been passed up from $\Lambda = 2$ to $\Lambda = 1$ and thence to $\Lambda = 0$, that t_0 has changed, or lastly c) control passes up to $\Lambda = 0$ at some time α equal to the length of a loop at $\Lambda = 1$, thus for some ${}^1\Sigma = \alpha$ and then $(t_0)^{L_\alpha} \neq (t_0)^{L_{\alpha^{++}}}$.*

More generally: control passes at a stage from $\Lambda = k+1$ to $\Lambda = k$ only when $(t_0, \ldots, t_k)$ becomes unstable; in particular when either (a) calculating at $\Lambda = k+1$ sees that $(t_0, \ldots, t_k)$ is directly different from one ${}^0M_1$ to ${}^0M_2$; (b) information has just been passed up from $\Lambda = k+2$ to $\Lambda = k+1$ that $(t_0, \ldots, t_k)$ has changed, or lastly c) control passes up to $\Lambda = k$ at some time α equal to the length of a loop, thus at some ${}^1\Sigma$, and then $((t_0, \ldots, t_k))^{L_\alpha} \neq ((t_0, \ldots, t_k))^{L_{\alpha^{++}}}$.

Proof: We consider just the first paragraph, as the second is just a generalisation of it. Suppose α is a time when control passes from $\Lambda = 1$ to $\Lambda = 0$. Suppose the straightforward alternatives (a) and (b) fail. By the construction then, control only so passes back to $\Lambda = 0$ in (2) at $\Lambda = 1$, when we have a loop of a computation, $({}^1\zeta', {}^1\Sigma')$ say, performed at $\Lambda = 1$ in which, by Lemma 4.3, the values of t_1 were undefined or unstable. Control then passed back up at time $\alpha = {}^1\Sigma'$. But had t_0 been additionally stable beyond ${}^1\Sigma'$ to ${}^1\Sigma'^{++}$ we should have concluded by $\Upsilon(c)$ that t_1 *was* defined and stable in $({}^1\zeta', {}^1\Sigma')$ - by the survival argument. A contradiction, and thus $(t_0)^{L_\alpha} \neq (t_0)^{L_{\alpha^{++}}}$. QED

Lemma 4.8. *(i) Let $\nu \in E^1$. Then $\Lambda(e_0, \nu) > 0$.*
(ii) Our overall computation can never reach a final looping pattern.

Proof: (i) This is an instance of Cor.4.6.

(ii) Suppose otherwise and that (ν, σ) is the final looping period of the master program. Then $\Lambda(e_0, \nu) = \Lambda(e_0, \sigma) = 0$. Then ν is a Σ_2-extendible, (as we have bookkept all the theories T_α) but it must be at least a limit of such by the definition of the program. Thus $\nu \in E^1$. This contradicts (i). QED

4.1 Applications

(I) Σ^0_3-determinacy (and $\Im\Sigma^0_3$-monotone inductive definitions) *cf.* [18].

(II) Π^1_2-Monotone Induction. *cf.* [4].

The theorem here is:

Theorem 4.9. *TFAE:*

(i) β is the least ordinal that supports an ω-nesting;

(ii) $\mathcal{P}(\omega) \cap L_\beta$ is the least β-model of Π^1_2-Monotone Induction;

(iii) Any strategy for a Σ^0_3-game is definable (in fact Δ_2-definable) over L_β;

(iv) Any ittm generalised E-recursive computation on integers, when not convergent, diverges by stage β.

Note: Hachtman [4] interpolated $(i) \Rightarrow (ii)$ and $(ii) \Rightarrow (iii)$.

Note: With α_0 the least so that $L_{\alpha_0} \prec L_\beta$, as a corollary to these arguments we have:

Theorem 4.10.

(i) For any Σ^0_3-game, in which Player I wins, there is a winning strategy which is Σ_1-definable over L_{α_0}; and hence the complete $G\Sigma^0_3$ is $\Sigma_1(L_{\alpha_0})$;

(ii) The ittm generalised E-computable reals are precisely those of L_{α_0};

(iii) The following are recursively isomorphic:

(a) the complete $G\Sigma^0_3$ -set; (b) the Σ_1-theory of L_{α_0}; (c) the E-halting set $H_{\mathsf{E}} = \{e |\ P^{\mathsf{E}}_e(e)\downarrow\}$.

Questions and Discussions:

- Is there any substantial difference between defining prewellorderings on the E-semidecidable sets through the use of the total length function H, or through the rank function Definition 2.2?

- ω-nestings of Σ_n-extendibles? ($L_\xi \prec_{\Sigma_n} L_\tau$)

- Let β_n be the least level of L on which can be based an ω-model supporting an ω-depth nesting of Σ_n-extendibles.

Q Characterise β_n? Is it the least level of the L-hierarchy over which strategies for n-Boolean combinations of Σ^0_3 sets are definable?

[Recall: Σ_{n+2}-KP (but not Σ_{n+1}-KP) proves $\mathrm{Det}(n\text{-Boolean}(\Sigma^0_3))$ ("1-Boolean$(\Sigma^0_3) =_{\mathrm{df}} \Sigma^0_3$. "$n$-Boolean$(\Sigma^0_3)$" is n-fold Boolean combinations of Σ^0_3).

For $n = 1$ see [17]; for $1 < n < \omega$ see Montalban-Shore [13].]

Q Is there a generalised higher type Σ_n-ittm recursion theory along the lines here, but for $n > 2$? *cf.* [3]. (Conjecture: Yes, just copy what is here.)

Q Spector-Gandy Theorem for generalised ittm recursion theory. (Conjecture: Yes.)

Q Characterise the *superjump* for generalised ittm recursion theory.

Q Does the class of semi-decidable in K sets have the Scale Property? (It has the PWO-property by the above.)

Bibliography

[1] K.J. Barwise. *Admissible Sets and Structures*. Perspectives in Mathematical Logic. Springer Verlag, 1975.

[2] S-D Friedman and P.D. Welch. Two observations regarding Infinite Time Turing machines. In I. Dimitriou, editor, *Bonn Interational Workshop on Ordinal Computability*, pages 44–48, Bonn, 2008. Hausdorff Centre for Mathematics, University of Bonn.

[3] S-D. Friedman and P.D. Welch. Hypermachines. *Journal of Symbolic Logic*, 76(2):620–636, June 2011.

[4] S. Hachtman. Determinacy and monotone inductive definitions. *Israel Journal of Mathematics*, 230(1):71–96, 2019.

[5] J.D. Hamkins and A. Lewis. Infinite time Turing machines. *Journal of Symbolic Logic*, 65(2):567–604, 2000.

[6] P. Hinman. *Recursion-Theoretic Hierarchies*. Ω Series in Mathematical Logic. Springer, Berlin, 1978.

[7] H.J. Keisler. *Model theory for Infinitary Logic*, volume 52 of *Studies in Logic and the Foundation of Mathematics,*. North-Holland Publishing Co., Amsterdam, London, 1971.

[8] S.C. Kleene. Recursive quantifiers and functionals of finite type I. *Transactions of the American Mathematical Society*, 91:1–52, 1959.

[9] S.C. Kleene. Turing-machine computable functionals of finite type I. In *Proceedings 1960 Conference on Logic, Methodology and Philosophy of Science*, pages 38–45. Stanford University Press, 1962.

[10] S.C. Kleene. Turing-machine computable functionals of finite type II. *Proceedings of the London Mathematical Society*, 12:245–258, 1962.

[11] S.C. Kleene. Recursive quantifiers and functionals of finite type II. *Transactions of the American Mathematical Society*, 108:106–142, 1963.

[12] P. Koepke. Turing computation on ordinals. *Bulletin of Symbolic Logic*, 11:377–397, 2005.

[13] A. Montalbán and R. Shore. The limits of determinacy in second order number theory. *Proceedings of the London Mathematical Society (3)*, 104(2):223–252, 2012.

[14] Y.N. Moschovakis. *Descriptive Set Theory*. Studies in Logic series. North-Holland, Amsterdam, 1980.

[15] P.D. Welch. Eventually Infinite Time Turing degrees: infinite time decidable reals. *Journal of Symbolic Logic*, 65(3):1193–1203, 2000.

[16] P.D. Welch. Characteristics of discrete transfinite Turing machine models: halting times, stabilization times, and normal form theorems. *Theoretical Computer Science*, 410:426–442, January 2009.

[17] P.D. Welch. Weak systems of determinacy and arithmetical quasi-inductive definitions. *Journal of Symbolic Logic*, 76(2):418–436, June 2011.

[18] P.D. Welch. $G_{\delta\sigma}$-games. Preprint Series NI-12050, Isaac Newton Institute, Cambridge, July 2012.