Transfinite Machine Models

P.D. Welch

September 28, 2011

1 Introduction

In recent years there has emerged the study of discrete computational models which are allowed to act *transfinitely*. By 'discrete' we mean that the machine models considered are not analogue machines, but compute by means of distinct *stages* or in *units of time*. The paradigm of such models is, of course, Turing's original machine model. If we concentrate on this for a moment, the machine is considered to be running a program P perhaps on some natural number input $n \in \mathbb{N}$ and is calculating $P(n)$. Normally we say this is a successful computation if the machine halts after a finite number of stages and we may read off some designated form of output: '$P(n)\downarrow$'. However if the machine fails to halt after a finite time it may be exhibiting a variety of behaviours on its tape. Mathematically we may ask what happens 'in the limit' as the number of stages approaches ω. The machine may of course go haywire, and simply be rewriting a particular cell infinitely often, or else the Read/Write head may go 'off to infinity' as it moves inexorably down the tape. These kind of considerations are behind the notion of 'computation in the limit' which we consider below.

Or, it may only rewrite finitely often to any cell on the tape, and leave something meaningful behind: an infinite string of $0, 1$'s and thus an element of Cantor space $2^{\mathbb{N}}$. What kind of elements could be there? Considerations of what may lay on an output tape at an infinite stage first surface in the notion of 'computation in the limit' or 'limit decidable'. Whilst the first publication on the matter seems to be two papers coincidentally appearing in the same year, 1965, as Martin Davis has commented, surely this was already known to Post?

Definition 1 (Putnam [59]) *P is a* trial and error predicate *if and only if there is a general recursive function f such that for every $x_1, \ldots, x_n$:*

$$\begin{aligned} P(x_1, \ldots, x_n) &\equiv \lim_{y\to\infty} f(x_1, \ldots, x_n, y) = 1 \\ \neg P(x_1, \ldots, x_n) &\equiv \lim_{y\to\infty} f(x_1, \ldots, x_n, y) = 0. \end{aligned}$$

Then it is possible to write out conditions for P holding that are Δ^0_2. Moreover any property that can be expressed in a Δ^0_2 way, can be expressed also in this form. However there can be no decidable function $g(m_1, \ldots, m_n)$ which tells us how long

one must wait for the value to have settled down: if such existed then one may easily see that the relation P becomes decidable..

By allowing ourselves to survey the whole tape 'at stage ω' we are performing a so-called '*super-task*' which is generally conceived to be a process which is brought to completion, but involves infinitely many stages of activity. If we allow supertasks into computational models there might be some interesting phenomena to discover. For the Turing machine model the whole of the tape is then usable, both for input strings as well as output strings. We then have essentially a computation at a *higher type* and such were much studied following Kleene's papers [35], [38], [37], [36] and [39] in the 1960's and early 1970's; the work here of Aczel, Gandy, Moschovakis and others, would eventually lead to the *theory of inductive definitions,* and the abstract theory of *Spector classes*, and to the theory of *admissible sets* in the work of Kripke, Platek, Barwise. Another strand of thought, under the influence of Kreisel and Sacks, led to *meta-recursion theory*, and ultimately *α-recursion theory* where the objects of computation were not numbers but ordinals (see [64] for an account of this latter development).

We should like to see how the theory of transfinitely running discrete computational machine models fits into this broader, older, more abstract theory. The theory of Infinite Time Turing Machines (ITTM) of Hamkins and Kidder (originating in the 90's but first published in [21] in 2000) lays down an attractive formulation for this, in particular for defining actions at limit stages of time. Previously one might have considered standard computational machines 'stacked up' or computations repeated using one infinite stage output recycled as the next input *etc.* This might be done either conceptually, or else thought of as inhabiting a particular spacetime (such as in the work of Etesi-Neméti and Hogarth) that allow - in a particular sense - for supertasks. The ITTM's provide a suitable yardstick for measuring or modelling the capabilities of other machines or arrangements.

Perhaps surprisingly *Infinite Time Register Machines* (ITRM's) which again keep the usual hardware of a finite number of registers, with numerical contents, lead to surprisingly strong computational models. In contradistinction to the finite case where the capabilities of Turing machines and register machines are the same, in the transfinite realm they markedly diverge.

Omitted in this chapter is any discussion of TM's or other machines in particular physical settings or implementations. Thus we have not discussed the models of Beggs and Tucker [3] that compute sets of numbers by some kinematic based device, or the models of Davies [12] that attempt to run a supertask within a fragment of Newtonian mechanics by means of accelerating machine components of exponentially decreasing size *etc.*, *etc.* These are essentially only standard Turing machines in a particular format or physical setting, so we have decided to exclude these. Also, as we have discussed the capabilities of machines in Malament-Hogarth spacetimes elsewhere [81], this is not included here. More seriously we omit any discussion of finite automata acting on, say infinite graphs, or infinite binary trees. This would have fitted well here.

Rather we emphasise here the logico-mathematical aspects rather than the physical. Since we are often considering runs of machines along countable wellorders, a certain amount of analysis or second order number theory is needed to ensure such ordinals exist. We have to hand the methodology of reverse mathematics with which to discuss the strengths of the various assertions that certain machines loop or halt in certain ordinal times *etc.* This we do where possible. In Section 2 we very briefly sketch connections with Kleene's seminal papers in higher type recursion theory. There the theory of *Kleene Recursion* on sets of reals is outlined, with its connections to hyperarithmetic sets, Borel sets and so forth. This then is expanded to include definitions of Spector Class, and hyperdegrees. This we believe is the right picture to have before one, when thinking about the Infinite Time Turing Machine model (Sect. 4) of Hamkins and Kidder. Here we explore this formalism in quite some detail, and express the classes defined within Δ^1_2 as Spector classes.

Since these machine models require transfinite ordinals to compute along the question arises as to what ordinals, and how strong a theory is needed to justify their existence. At a couple of points below we illustrate in terms of traditional theories such as $\mathsf{\Pi^1_1\text{-}CA_0}$ and subsystems of $\mathsf{\Pi^1_3\text{-}CA_0}$ that play a role. This is in the spirit of the reverse mathematics of [70].

We set $\mathcal{L}_{\dot\in}$ to be the language of set theory. By ZF^- we mean the axiom system of Zermelo-Fraenkel with the Axiom Scheme of Collection, but the Axiom of Power Sets removed. By $\mathsf{\Sigma_n\text{-}KP}$ we mean ZF^- but with Collection restricted to Σ_n, and Separation to Δ_n, formulae respectively. ($\mathsf{\Sigma_1\text{-}KP}$ is then just KP.) When notions are used from ordinary Turing computability theory and there is danger of confusing them with some new broader notion under discussion we refer to them as 'standard'. By '$A \leq_1 B$' for $A, B \subseteq \mathbb{N}$ we mean that $A = f^{-1}\text{“}B$ for a total (1-1) computable (*i.e.* standard) function f . By '$A \equiv_1 B$' we mean that $A = f^{-1}\text{“}B$, for a (standard) computable bijection $f : \mathbb{N} \to \mathbb{N}$, in other words that A is computably isomorphic to B. $A \leq_T B$ is the standard Turing reducibility. We shall use notions of 'Γ *being a pointclass*' of either a set of integers, or sets of reals - the latter will be mostly thought of as coextensive with $2^{\mathbb{N}}$ (but sometimes $\mathbb{N}^{\mathbb{N}}$, however it will be clear which), in the sense of Moschovakis [56]. We shall not have any occasion to consider Polish spaces other than products of these two, and the latter reference will contain all the descriptive set-theoretic definitions needed for Spector classes, Uniformization, Scales and the like. Everything we do will take place within the pointclass of Δ^1_2. For an account of recursive-theoretic herarchies within this pointclass, see [31]. We let WO denote the set of real numbers coding wellorderings. For λ a limit ordinal and $\langle l(\alpha) \mid \alpha < \lambda\rangle$ a sequence of natural numbers, by $Liminf^*\langle l(\alpha) \mid \alpha < \lambda\rangle$ we mean the usual $Liminf\langle l(\alpha) \mid \alpha < \lambda\rangle = sup_{\beta<\lambda}\big(inf\{l(\alpha) \mid \alpha > \beta\}\big)$, but only if this is less than ω. If the latter equals ω, then the $Liminf^*$ operation returns 0.

2 Kleene's Equational Calculus for higher types

In a series of papers ([35] - [38]) Kleene developed a generalisation of his earlier equational calculus for the Gödel-Herbrand general recursive functions from the 30's. In this generalisation *higher-type recursion* was developed. This generalized theory saw numbers as objects of *type* - 0 a function $f : \mathbb{N}^k \to \mathbb{N}$ as an object of *type* 1, and $F : (2^{\mathbb{N}})^k \times \mathbb{N}^l \to \mathbb{N}$ as of *type*-2, and so forth. (See the account of Normann [57] in this volume.) This was continued into further types. It is not the purpose of this article to describe in any great details those theories but concentrating only on the type-1 and type-2 cases. One notion of computation can be given a mechanistic flavour, and illustrated by what Rogers [61] (attributing the phrase to Spector and Addison) called the "$\aleph_0$-mind" . One imagines a Turing machine with a genuinely infinite tape of cells $\langle C_i \mid i < \omega \rangle$, but with the capability of surveying and manipulating a countable amount of information in a single step. The machine is deemed capable of consulting an oracle that gives yes/no answers to queries of the form *Is* $y \in A$*?* where $y \in 2^{\mathbb{N}}$ is the tape's contents at any stage of computation, and $A \subset \mathbb{N} \cup 2^{\mathbb{N}}$. The machine must thus both be able to calculate the values of $y(k)$ to complete y, and be thought of as being able to present y to the oracle for query. Computations are thus truly infinitary objects and indeed a successful computation is best thought of as represented by a wellfounded tree of computation steps and subroutine calls.

Such a tree has *infinite branching* below any node (perhaps representing the calculations $y(0), y(1) \ldots$ *etc*). The output of such a machine could then be either integer or real form.

Generalising to several variables $\vec{n}$ of number type and $\vec{x}$ of real type, we may index programs of such machines by $e \in \omega$ and write a typical such calculation with oracle $\mathcal{I}$ as $\{e\}(\vec{n}, \vec{x}, \mathcal{I})$. Recursion in ${}^2E$ is essential for much of the development of relative recursiveness for the computation theory at this type, where, for $\alpha \in 2^{\mathbb{N}}$:

$$\mathcal{E}(x) = \begin{cases} 0 \text{ if } \exists n x(n) = 0 \\ 1 \text{ otherwise.} \end{cases}$$

For any oracle $\mathcal{T}$ the class of relations semi-decidable in $\mathcal{T}$ is closed under universal number quantification; by requiring that computations be relative to $\mathcal{E}$ we also ensure their closure under existential number quantification (as the form of $\mathcal{E}$ shows). This then implies that any arithmetical sets of reals is Kleene-computable. However requiring computations to be relative to $\mathcal{E}$ also ensures that we have an *Ordinal Comparison Theorem* ; this is needed to develop the theory of relations semi-computable in a type-2 functional. Kleene then showed that the decidable relations were precisely the hyperarithmetic, and the semi-decidable are those 'Kleene reducible' to a complete Π^1_1 set of reals, say WO, that of the codes of countable wellorders.

Theorem 1 (Kleene) *The hyperarithmetic relations $R(\vec{n}, \vec{x}) \subseteq \mathbb{N}^k \times (\mathbb{N}^{\mathbb{N}})^l$ for any $k, l \in \mathbb{N}$.are precisely those computable in $\mathcal{E}$.*

The Π^1_1 relations are precisely those semi-computable in $\mathcal{E}$.

One may define the notion of *Kleene degree* from the notion of *Kleene reducibility*:

Definition 2 ***Kleene reducibility:*** *Let $A, B \subseteq \mathbb{R}$; we say that $A \leq_K B$ iff there is an index e and $y \in \mathbb{R}$ so that*

$$\forall x \in \mathbb{R}(x \in A \longleftrightarrow \{e\}(x, y, B, \mathcal{E}) \downarrow 1);$$

$$\forall x \in \mathbb{R}(x \notin A \longleftrightarrow \{e\}(x, y, B, \mathcal{E}) \downarrow 0).$$

A is Kleene-semi-computable in *B iff there is an index e and $y \in \mathbb{R}$ so that*
$$\forall x \in \mathbb{R}\ (x \in A \longleftrightarrow \{e\}(x, y, B, \mathcal{E}) \downarrow 1)).$$

Notice that the obvious degree notion has become a boldface one: we are allowed a real parameter y into the definition. We may if we wish think of y as a variable adding some countable amount of information to the computation, but in any case the bottom-most degree $\mathbf{0_K}$, say, containing $\varnothing, \mathbb{R}$, in fact now consists of the *Borel sets*. The semi-computable sets are now those of degree precisely the co-analytic $\mathbf{\Pi^1_1}$ sets, In terms of a jump notation the Kleene degree of a complete Π^1_1 such as WO, may be written $\mathbf{0'_K}$. Then the degree of $A \subseteq \mathbb{R}$ contains continuum many sets $B \subseteq \mathbb{R}$. The presence of the parameters y ensures that the degree is closed under continuous pre-images, and hence is a *Wadge degree*.

The notion of Kleene degree here is then tied in very much with hyperarithmeticity and WO. It is thus a finer degree notion than that of Δ^1_2-degree. We may formulate a general slice through the Δ^1_2-pointclass as follows. Suppose $f : \mathbb{R} \longrightarrow \omega_1$ is such that $x \leq_T y \longrightarrow f(x) \leq f(y)$. The function f is then *Turing invariant*. Further let us suppose f is Σ_1 definable over $(\mathrm{HC}, \in)$ without parameters, by a formula in $\mathcal{L}_{\dot{\in}}$.

Definition 3 *Let f be as described; let Φ be a class of formulae of $\mathcal{L}_{\dot{\in}}$. Then $\Gamma = \Gamma_{f,\Phi}$ is the pointclass of sets of reals A so that $A \in \Gamma$ if and only if there is $\varphi \in \Phi$ with:*

$$\forall x \in \mathbb{R}(x \in A \leftrightarrow L_{f(x)}[x] \models \varphi[x])$$

The idea is that we are allowed, for example if $\Phi = \Sigma_1$, to only search through the $L[x]$ hierarchy as far as $f(x)$ for a witness to show that $\varphi[x]$ holds. Allowing f and Φ to vary then carves out a slice through the Δ^1_2 sets. As an example, with $\Phi = \Sigma_1$ and $f(x) = \omega^x_{1,\mathrm{ck}}$ (the first ordinal that is not x-recursive) this yields the Π^1_1 sets; allowing an arbitrary $y \in \mathbb{R}$ as a fixed parameter to go with a choice of $\varphi \in \Sigma_1$ would yields the $\Pi^1_1(y)$ sets. Hence Def. 3 can be relativised uniformly to any parameter.

As is well known $L_{\omega^z_{1,\mathrm{ck}}}[z]$ is the least transitive model of Kripke-Platek set theory containing ω and z; it is thus the "least z-admissible" ordinal (*cf.* [1],V 5.11). We may extend this form and define $\omega^{B,y}_{1,\mathrm{ck}}$ to be the ordinal rank of the least B-y-admissible set, now in a language extended with a predicate for the set of reals B. The relation $A \leq_K B$ of Def.2 can be usefully reformulated.

Lemma 1 *$A \leq_K B$ iff there are Σ_1-formulae in $\mathcal{L}_{\in,\dot{X}}$ $\varphi_1(\dot{X}, v_0, v_1), \varphi_2(\dot{X}, v_0, v_1)$, and there is $y \in \mathbb{R}$, so that*

$$\begin{aligned} \forall x \in \mathbb{R}(x \in A &\iff L_{\omega_1^{B,y,x}}[B, y, x] \models \varphi_1[B, y, x] \\ &\iff L_{\omega_1^{B,y,x}}[B, y, x] \models \neg\varphi_2[B, y, x]). \end{aligned}$$

Similarly A would be Kleene-semi-computable in B if only if there is a formula φ_1 as above with just the first line holding (see [32]).

Solovay showed that, assuming Axiom of Determinacy, the ordering of the Kleene degrees under strict Kleene-reducibility was wellordered.

The structure of the Kleene degrees of sets of reals is very much dependent on the ambient set-theoretical universe: Solovay observed ([71]) that if the Axiom of Determinacy holds then $\leq_K$ is ordered; if $V = L$ (or set forcing extensions thereof) then there are intermediate degrees $\mathbf{0_K} <_K \boldsymbol{B} <_K \mathbf{0'_K}$ (indeed $2^{\mathfrak{c}}$ many mutually incompatible such) Hrbacek-Simpson,[32]); however if $\mathrm{Det}(\mathbf{\Pi^1_1})$ holds then there are none (Harrington [29]).

Definition 4 (Moschovakis [56]) *A* Spector class *of pointsets $\Gamma \subseteq \mathbb{N}^k \times (\mathbb{N}^{\mathbb{N}})^l$ for any k, l, is a collection that is (i) closed under $\cap, \cup$, number quantification $\exists^{\mathbb{N}}$, $\forall^{\mathbb{N}}$; closed under (standard) recursive substitutions, has a universal set U indexing by $\mathbb{N}$ all members of Γ, and lastly has the Prewellordering property:*

PW: For any $P \in \Gamma$ there is $\sigma : P \longrightarrow \lambda$ for some ordinal λ with the property that there are relations: $x \leq^\sigma_0 y \in \Gamma$, $x \leq^\sigma_1 y \in \breve{\Gamma}$ so that:

$$P(y) \Rightarrow \forall x([P(x) \wedge \sigma(x) \leq \sigma(y)] \iff x \leq^\sigma_0 y \iff x \leq^\sigma_1 y).$$

The paradigm here is the class of Π^1_1 sets itself: as any such $P \in \Pi^1_1$ has the form: $P = \{y \mid L_{\omega^y_{1,\mathrm{ck}}}[y] \models \varphi[y]\}$ for some Σ_1 φ, we may define $\sigma(y) \simeq \mu\alpha.L_\alpha[y] \models \varphi[y]$, and this is a suitable prewellordering. We then may set

$x \leq^\sigma_0 y \iff \exists \eta < \omega_1^{x,y}(L_\eta[x, y] \models \sigma(x) \leq \sigma(y)])$;

$x \leq^\sigma_1 y \iff \forall \eta < \omega_1^{x,y}(L_\eta[y] \models \varphi[y] \longrightarrow L_\eta[x] \models \varphi[x])$

As relations $\Pi^1_1(x, y)$ are precisely those $\Sigma_1([L_{\omega_1^{x,y}}[x, y])$ (essentially by the Spector-Gandy Theorem, [72], [20]) we have relations of the right complexity. (The existence of a universal set U can be justified by the existence of a universal Σ_1 formula and Skolem function in $\mathcal{L}_{\dot{\in}}$ for models of "$V = L[\dot{z}]$".). Other Spector classes are the projective classes $\Sigma^1_k (k \geq 2)$ assuming $V = L$, or the classes Π^1_{2n+1}, and Σ^1_{2n+2} $(n \geq 0)$ under Projective Determinacy (PD *cf.* [56]), but there are many others.

We shall be deriving some from the transfinite machine model, analogously to that obtained from Kleene reducibility, which we shall describe below.

The notions above for sets of integers are clearly closely related to that of hyperarithmeticity between sets of integers, and hyperdegrees where '$x \leq_h y$' is the reducibility of the set x is hyperarithmetic in the set y. For such we have a so-called *Spector Criterion for hyperdegrees:*

$$x \leq_h y \longrightarrow (x' \leq_h y \longleftrightarrow \omega_{1,\mathrm{ck}}^x < \omega_{1,\mathrm{ck}}^y)$$

where x' is the complete $\Sigma_1^1(x)$ set of integers - the *hyperjump* of x. We shall see that other reducibilities related to transfinite computational models have an associated jump operation, and can have ordinal assignments $x \rightarrowtail \lambda^x$ which satisfy the analogous form of criterion.

3 Infinite Time Turing Machines

We turn now to considering the behaviour of ordinary machines, those designed by Turing for the analysis of computability in the ordinary sense, but allowing them to have transfinite amounts of time or even space in which to run.

3.1 Basic properties

We shall give a somewhat historically ordered account of these over the last decade, starting with the description of the Infinite Time Turing Machines (ITTM's) due to Hamkins and Kidder. These were considered in the 1990's but only appeared in print in an extensive account by Hamkins and Lewis in 2000, [21]. (Hamkins tells the story that he and Kidder invented the ITTM in their office directly after hearing Leonore Blum speak on the Blum-Shub-Smale machines at the Berkeley Logic Colloquium.) This paper awakened interest in such transfinitely running models. Later others turned to Register machines, or Turing machines with ordinal length tapes, thus increasing the space component of such machines.

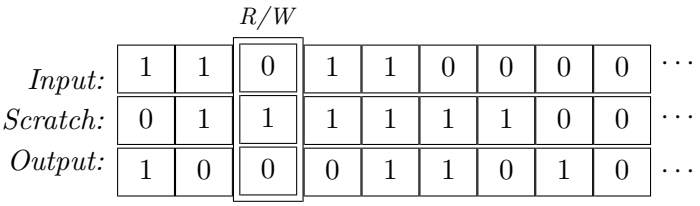

Figure 1: A 3-tape Infinite Time Turing Machine

We shall look at the mathematics of the ITTM model in detail first, before turning to these others. Such a machine is essentially a standard Turing machine, with a one way infinite tape of cells $\langle C_i \mid i \in \mathbb{N} \rangle$, which we consider divided up into

three infinite tapes for *Input, Scratch,* and *Output* tapes, say as $\langle C_{3i+j} \mid i \in \mathbb{N}\rangle$ for $j < 3$ for definiteness. In [21] a special *limit state*, q_L, was added to the already finite number of states of the machine $q_0, \ldots, q_N$ say. This was the state the machine entered into at a limit stage of time, but which we shall dispense with here. The program software of the machine with its finite transition state table is thus unchanged: programs of the standard machine $\langle P_e \mid e \in \mathbb{N}\rangle$ can be carried over to this transfinite context. All we have to do is specify a behaviour at limit stages λ of time. Firstly we by decree define that the cells' values, $C_i(\lambda)$, are determined by a final segment of the values $C_i(\alpha)$ at ordinal times $\alpha < \lambda$:

$$\begin{aligned} C_i(\lambda) &= k \text{ if } \exists \alpha < \lambda \forall \beta < \lambda (\alpha < \beta \longrightarrow C_i(\beta) = k) \text{ for } k \in \{0,1\}; \\ &= 0 \text{ otherwise.} \end{aligned}$$

This 'limit rule' is thus a *Liminf*, and is of an $\exists\forall$ nature. (The original paper used *Limsup*'s instead but this is an inessential difference.) Having specified the cell values, if we denote the state of the machine at time α as $q(\alpha)$, and the position the R/W head is at as $l(\alpha)$, then we need only to specify the values of $q(\lambda)$ and $l(\lambda)$. We again use a liminf rule and specify:

$$\begin{aligned} q(\lambda) &= Liminf\langle q(\alpha) \mid \alpha < \lambda\rangle \\ l(\lambda) &= Liminf^*\langle l(\alpha) \mid \alpha < \lambda\rangle \end{aligned}$$

The reader may like to verify that the upshot is that if we set out our Turing machine transition table as lines of a program with suitably nested subroutine calls and so forth in a sensible fashion, then matters can be arranged so that then the machine is placed back in the state/instruction at the beginning of the outermost subroutine in the program that was called unboundedly often before time λ; moreover the head is then also placed at that cell from which it entered into that subroutine, again unboundedly often before λ. This is a pleasing but inessential detail in the arrangement. (The original paper had the machine enter the special limit time state q_L and the R/W head always went back to the beginning of the tapes.)

We now have defined the behaviour of the machine for any ordinal stage of time, for any sequence $x \in 2^{\mathbb{N}}$ that may be written on the input tape at the outset. Note that computations with oracles $Z \subseteq \omega$ are rendered superfluous, since Z may be written to the input tape, and the whole tape can be scanned in ω many steps. We can thus think now of oracles as being sets $A \subseteq 2^{\mathbb{N}}$, and we may quiz the oracle as to whether, *e.g.* , the contents of the scratch tape is in A. We thus shall have a model of computations in a higher type. Hamkins and Lewis explore the possibilities of this model, mostly emphasising the analogy with Turing computations, forming notions of relative computability, computation degrees and the like. Given the affinity with the standard model one sees that one has many of those features; they prove a wealth of features which we cannot list all here, but they include the existance of a *universal machine*, there are S^n_m and *Recursion Theorems*. To give a flavour of several of the proofs in [21] we prove:

Theorem 2 (Hamkins-Lewis - [21]) *(i) '$x \in WO$' is decidable by an ITTM, where WO is the set of reals that are codes of wellorders. (ii) Moreover let α be any recursive ordinal and $A \subseteq 2^{\mathbb{N}}$ an α-Π^1_1 set, that is, at the α'th level of the difference hierarchy on Π^1_1 sets. Then '$x \in A$' is decidable by an ITTM.*

Proof (i) We do this just for the single Π^1_1 set The proof for (ii) at the higher levels is just complication. It can be quickly established that such machines will be able to decide whether an input real codes a wellorder: note first that any arithmetic predicate is decidable: one may even decide a Σ^0_2 predicate of integers in ω steps: to ask if $x \in A$ with $A \in \Sigma^0_2$, defined as $\exists u \forall v \Upsilon(u, v, x)$ with Υ in Δ_0, one computes of $\Upsilon(0, 0, x), \Upsilon(0, 1, x) \ldots$ in a standard Turing manner, until, if ever, a k is found so that $\neg\Upsilon(0, k, x)$, if so then a 'flag cell', say C_1 has its value switched $1 \to 0 \to 1$, and then one proceeds with $\Upsilon(1, 0, x), \Upsilon(1, 1, x) \ldots$ *etc.* Then $x \in A \leftrightarrow C_1(\omega) = 1$. (In a further ω many steps we may decide Σ^0_4 predicates, and thus by time ω^ω any arithmetic predicate.) As being a linear order is Π^0_2, the machine may decide if x codes such a linear order, and thereafter seeks for the least element in the ordering coded by x. It makes a guess by taking an element from the field of x, k_0 say, and starts searching for a lesser element k_1, if such exists, then it is found in finitely many steps; repeating a search for k_2 *etc*; if each time a lesser element is found we can flash a flag cell $1 \to 0 \to 1$, then after ω many stages if the flag is zero, then we found an infinite descending chain, and the ordering is illfounded; if it is 1, then there is some value k_m that was $<_x$-minimal. In ω further steps we go through the field of the wellorder, erasing all mention of k_m; we then pick another guess as a new k'_0 and repeat. Eventually we either erase the whole of Field($<_x$) or we find an infinite descending chain, and in the former case we know we have a wellorder.

Q.E.D.

They show that even further levels of the difference hierarchy on Π^1_1 can be decided by such programs, in fact for α that are *writable*, in a sense to be made precise below. We may denote by '$P_e(x) \downarrow y$' that y is the output of a halting computation of a machine on program P_e with input x. We may input an integer n by a string of $n+1$ 1's followed by 0's, but we abbreviate this by writing '$P_e(n) \downarrow y$'. If the computation fails to halt then we write '$P_e(x) \uparrow$'.

Definition 5 *(i) We write '$P_e(n) \downarrow^\alpha y$' if $P_e(n) \downarrow y$ in exactly α steps. We call α* clockable *if $\exists e \exists n \in \omega \exists y\ P_e(n) \downarrow^\alpha y$.*

(ii) A real $y \in 2^{\mathbb{N}}$ is writable *if there are $n, e \in \omega$ with $P_e(n) \downarrow y$; an ordinal β is called writable, if β has a writable code y.*

An ITTM calculation such as that of $P_e(n)$ can of course now halt in a transfinite number of steps, and it is amusing to think of programs that halt *precisely* in ω, $\omega^2, \omega^\omega.2$ steps. An elementary argument shows that if we regard a *snapshot* of such a machine at time α as consisting of $\langle l(\alpha), q(\alpha), \langle C_i(\alpha)\rangle_i\rangle$ then a non-halting program must eventually after a countable number of steps enter into some permanently repeating loop. (One should note that it is not sufficient to merely have a pair of

identical snapshots in order to declare that a computation is in a permanent loop: it is necessary to have a closed increasing $\omega+1$-sequence of ordinals $\gamma_i (i \leq \omega)$ with identical snapshots for this: even an ω-sequence $\langle \delta_i \rangle_i < \omega$ may result in $C_k(\delta_i) = 1$ whilst having $C_k(\sup\{\delta_i\}) = 0$) by the *liminf* formulation. It can be shown however for the universal machine that any repeating pair of snapshots will be permanently repeating.) We thus see that the property of being a '*well-ordered sequence of snapshots in the computation* $P_e(x)$*'* is Π^1_1 as a relation of e and x; thus '$P_e(x)\downarrow y$' is Δ^1_2:

$\exists w$(w *codes a halting computation of* $P_e(x)$*, with* y *written on the output tape at the final stage*) $\iff$

$\forall w$(w *codes a computation of* $P_e(x)$ *that is either halting or performs a repeating infinite loop* $\longrightarrow$ w *codes a halting computation with* y *on the output tape*.)

Similarly $P_e(x)\uparrow$ is also Δ^1_2. In [21] the authors conduct an analysis of *gaps* in the clockable ordinals and prove a number of results, including: the first maximal gap in which no computation of the form $P_e(n)$ halts is $[\omega_1^{\rm ck}, \omega_1^{\rm ck} + \omega)$; further no calculation halts in precisely an admissible β number of steps; no gap is shorter than ω but there are arbitrarily long gaps appearing cofinally below the supremum γ of clockable ordinals.

Definition 6 *(i)* $x^\nabla = \{e | P_e(x)\downarrow\}$ *(The halting set on integers).*
(ii) $X^\blacktriangledown = \{(e,y) | P_e^X(y)\downarrow\}$ *(The halting set on reals relativised to* $X \subseteq 2^{\mathbb{N}}$*).*

It is natural to ask for characterisations of, *e.g.* 0^∇, or to ask: what are the clockable ordinals? Before turning to this we define:

Definition 7 *(i)* $R(x)$ *is an* ITTM-semi-decidable *predicate if there is an index* e *so that:*

$$\forall x (R(x) \leftrightarrow P_e(x)\downarrow 1)$$

(ii) A predicate R *is* ITTM-decidable *if both* R *and* $\neg R$ *are* ITTM-*semi-decidable.*

Clearly one would like to know what the ITTM-(semi-)decidable sets are. To get a grasp on an upper bound note, at the risk of stating the obvious, that we have one clear difference between standard Turing computations and that of an ITTM: for the former a single integer suffices to code a whole *course of computation*, whereas to code a course of an ITTM-computation is to code a wellordered sequence of snapshots the form $\langle q(\alpha), l(\alpha) \langle C_i(\alpha) \rangle_{i<\omega} \rangle$ each of which may be coded as real. We might try to get an analogue of Kleene's canonical *Normal Form Theorem* arising from his T-predicate. For any halting standard Turing computation there is an integer K that will code the whole course of the computation $P_e(n)$, and we may effectively find an e' so that $\forall n (P_e(n)\downarrow \longrightarrow P_{e'}(n)\downarrow K)$. For this to work in the ITTM arena even for computations on integers, we must have that the *ordinal lengths* of

the computations are writable by computations. In short: *Is every clockable ordinal writable?*

Before answering this we remark that non-halting behaviour has two aspects: there can be a computation that, whilst not formally halting nevertheless continues without making any further changes to its output tape - it may merely alter entries to the scratch tape for ever; in one sense then the output has *stabilized* even if the machine has not formally halted. The other kind of non-halting behaviour results in the output tape being infinitely modified. This leads to:

Definition 8 *(i) Suppose for the computation $P_e(x)$ the machine does not halt then we write $P_e(x)\uparrow$; if eventually the output tape does have a stable value $y \in 2^{\mathbb{N}}$ then we write: $P_e(x)\uparrow y$ and we say that y is* eventually writable*.*

(ii) $R(x)$ is an eventually ITTM-semi-decidable *predicate if there is an index e so that:*

$$\forall x(R(x) \leftrightarrow P_e(x)\uparrow 1)$$

(iii) A predicate R is eventually ITTM--decidable *if both R and $\neg R$ are eventually* ITTM-*semi-decidable.*

There are concomitant questions as to what these predicates are, how long it takes for a computation to start looping, and if the output is eventually writable, after how many stages *etc.*, and indeed about halting or stabilizing behaviour in general. Resolving the question of whether all clockable ordinals were all writable (in short *halting* behaviour) turned out to require an analysis of the *stabilizing* behaviour of individual cells C_i during the course of a universal machine computation (*cf.* [78]). In this sense stabilization is prior or anterior to halting. Once stabilizing behaviour is fully analysed one has:

Theorem 3 (The $\boldsymbol{\lambda, \zeta, \Sigma}$-Theorem)(Welch cf. [77], [83]) *(i) Any ITTM computation $P_e(x)$ which halts, does so by time λ^x, the latter being defined as the supremum of the x-writable ordinals;*
(ii) any computation $P_e(x)$ with eventually stable output tape, will stabilize before the time ζ^x defined as the supremum of the eventually x-writable ordinals;
(iii) moreover ζ^x is the least ordinal so that there exists $\Sigma^x > \zeta^x$ with the property that

$$L_{\zeta^x}[x] \prec_{\Sigma_2} L_{\Sigma^x}[x];$$

(iv) then λ^x is also characterised as the least ordinal satisfying:

$$L_{\lambda^x}[x] \prec_{\Sigma_1} L_{\zeta^x}[x].$$

Considering just the integer computational version, by taking $x = \varnothing$ in the above, since the ITTM operations are simply defined and absolute, one sees that running the computation inside L, that if $P_e(n)$ is non-halting, then, with hindsight, the Σ_2 liminf rules will ensure that the snapshots at times $\zeta(= \zeta^{\varnothing})$ and $\Sigma(= \Sigma^{\varnothing})$

(taken as defined by the characterisation of (iii) above) are the same. It is then easy to argue that if the computation has not halted by time ζ then it will never halt but continue with periodicity Σ. Thus the lexicographically least pair $(\bar{\zeta}, \bar{\Sigma})$ standing in the relationship of *(iii)* of the theorem, forms an upperbound to the least pair of permanently repeating snapshots of the universal computation. (The reader may be concerned about the warning issued earlier that having merely a pair of ordinals (ζ, Σ) with identical snapshots is not sufficient to guarantee that we are in a permanently looping cycle. This is true, but note here no cell C_j that is stable on a tail interval $(\delta, \zeta]$ will change its value in $[\zeta, \Sigma)$: the first ordinal $\delta' > \delta$ where C_j changed its value (were it to exist) would be $\Sigma_2(L_\Sigma)$ definable, and hence (by the characterisation of (iii)) would be less than ζ. So actually any pair of snapshots of any computation on integers, at this particular pair of stages must be permanently cycling.) As a lower bound one can show that if one has a code for α on a tape one can use this to build a code for L_α on another.

Since λ $(= \lambda^{\varnothing})$ has the Σ_1-elementarity property above, and since '$P_e(n)\downarrow$' is a Σ_1 statement, we have immediately that any clockable ordinal is below λ. This, together with the observation from [21] that the 'counting through' algorithm of any code for a wellordering of order type τ takes at least τ many steps before halting, yields that any writable ordinal is also clockable. The corollary is that the two classes are the same. Hence, following Kleene:

Theorem 4 (Normal Form Theorem)(Welch) *(a) For any ITTM computable function φ_e we can effectively find another ITTM computable function $\varphi_{e'}$ so that on any input x from $2^{\mathbb{N}}$ if $\varphi_e(x)\downarrow$ then $\varphi_{e'}(x)\downarrow y \in 2^{\mathbb{N}}$, where y codes a wellordered computation sequence for $\varphi_e(x)$.*
(b) There is a universal predicate $\mathfrak{T}_1$ *which satisfies $\forall e \forall x$:*

$$P_e(x)\downarrow z \quad \leftrightarrow \quad \exists y \in 2^{\mathbb{N}}[\mathfrak{T}_1(e, x, y) \wedge Last(y) = z].$$

Further, as a corollary (to Theorem 3):

Corollary 1 *(i) $x^{\nabla} \equiv_1 \Sigma_1$-Th$(\langle L_{\lambda^x}[x], \in, x\rangle)$ - the latter the Σ_1-theory of the structure.*

(ii) Let $x^{\infty} =_{\mathrm{df}} \{e \mid \exists y P_e(x) \uparrow y\}$ be the set of x-stable indices. *Then*

$$x^{\infty} \equiv_1 \Sigma_2\text{-Th}(\langle L_{\zeta^x}[x], \in, x\rangle).$$

The ITTM-decidable sets of integers consist precisely of those in L_λ, and the ITTM-semidecidable sets are those $\Sigma_1(L_\lambda)$; both of these with the obvious uniform relativised counterparts. The ITTM-eventually decidable sets turn out then, to be those which are $\Delta_2(L_\zeta)$, and as L_ζ is a Σ_2-admissible set, are elements of L_ζ; and the ITTM-eventually-semi-decidable those that are $\Sigma_2(L_\zeta)$. Again their relativisations to real inputs are uniform. To see these latter claims from Cor.1(ii), note that if the n'th cell of the output tape of the machine running $P_e(0)$ is stable from some point

on, then this fact is expressed by a Σ_2 sentence about e and n over L_ζ; hence if all the cells of the output tape stabilize a simple application of Σ_2-admissibility of ζ shows that this will have happened for them all by a stage $\bar{\zeta} < \zeta$. Hence if $P_e(0) \uparrow y$ then $y \in L_\zeta$.

The last corollary should be compared with Kleene's result that the complete Π^1_1-set of integers coding indices of wellfounded recursive trees, that is the notation system $\mathcal{O}$ (see [61]), is recursively isomorphic to the Σ_1 theory of $\langle L_{\omega_1^{\rm ck}}, \in\rangle$. Indeed it is easy to see that one can literally extend Kleene's $\mathcal{O}$ to an $\mathcal{O}^+$ and also to an $\mathcal{O}^\infty$ by simply allowing indices into $\mathcal{O}^+$ of those programs that halt, and into $\mathcal{O}^\infty$ those of stable output: we use precisely the same formalism as for $\mathcal{O}$, and simply widen the definition of 'computable'. Thus $\mathcal{O}$ is to $L_{\omega_1^{\rm ck}}$ as $\mathcal{O}^+$ is to L_λ as $\mathcal{O}^\infty$ is to L_ζ (the details of this are in [40]). It is natural to regard both x^∇ and x^∞ as jump operators and later we shall refer to them as such.

Definition 9 *(i) A set of integers x is* semi-decidable *in a set y if and only if:*

$$\exists e \forall n \in x\, [P_e^y(n)\downarrow 1 \longleftrightarrow n \in x\ ]$$

(ii) A set of integers x is decidable *in a set y if and only if:*

$$\exists e \forall n \in x\, [(P_e^y(n)\downarrow 1 \leftrightarrow n \in\ x) \wedge (P_e^y(n)\downarrow 0 \leftrightarrow n \notin\ x)]\,.$$

We may write $x \leq_\infty y$ for the reducibility ordering.

(iii) A set of integers x is eventually-(semi)-decidable *in a set y if and only if the above holds with $\uparrow$ replacing $\downarrow$. For this reducibility ordering we write $x \leq^\infty y$.*

We thus compactly express the two reducibilities using the notations $\leq_\infty$ and $\leq^\infty$ with the first denoting computability, and the second eventual computability.

Lemma 2 *(i) The assignment $x \rightarrowtail \lambda^x$ satisfies the Spector Criterion:*

$$x \leq_\infty y \longrightarrow (x^\nabla \leq_\infty y \leftrightarrow \lambda^x < \lambda^y).$$

(ii) Similarly for the assignment $x \rightarrowtail \zeta^x$:

$$x \leq^\infty y \longrightarrow (x^\infty \leq^\infty y \leftrightarrow \zeta^x < \zeta^y)$$

The above shows that degree structure induced by $\leq_\infty$ is more akin to that of hyperdegrees than Turing degrees; indeed the proper analogy is probable more that with Δ^1_2-degrees; in terms of fineness, ITTM-reducibility and eventual-ITTM-reducibility are strictly in between these two. The following is the natural version for real computation.

Definition 10 *A set of reals A is* semi-decidable *in a set of reals B if and only if:*

$$\exists e \forall x \in 2^{\mathbb{N}}\, \left[P_e^B(x)\downarrow 1 \leftrightarrow x \in A\ \right]$$

(ii) A set of reals A is decidable *in a set of reals B if and only if:*

$$\exists e \forall x \in 2^{\mathbb{N}}\, \left[P_e^B(x)\downarrow 1 \leftrightarrow x \in A \wedge P_e^B(x)\downarrow 0 \leftrightarrow x \notin A\ \right]$$

(iii) If in the above we replace $\downarrow$ everywhere by $\uparrow$ then we obtain the notion in (i) of A is eventually decidable in B *and in (ii) of* A is eventually semi-decidable in B.

Definition 11
(i) $A \leq_\infty B$ *iff for some* $e \in \omega$, *for some* $y \in 2^{\mathbb{N}}$: A *is decidable in* (y, B).
(ii) $A \leq^\infty B$ *iff for some* $e \in \omega$, *for some* $y \in 2^{\mathbb{N}}$: A *is eventually decidable in* (y, B).

Again two reducibilities, one for each notion. Also a real parameter has been inserted into the last definition; this brings it into line with the previous notion of Kleene reducibility between sets of reals, and is closed under continuous pre-images, and thus again will ensure that each degree is a Wadge pointclass. Just as for Kleene degrees the structure of either of these induced degree orderings will depend on the ambient set theory: whether $V = L$ (and this implies there are many degrees, even incomparable degrees below the complete ITTM-(eventually)-semi-decidable set) or there is "sufficient determinacy". In the latter case the degrees will be wellordered and without any intermediate degrees at all. This is discussed further below.

By analogy with Kleene recursion we have:

Lemma 3
(i) $A \leq_\infty B$ *iff there are* Σ_1*-formulae in* $\mathcal{L}_{\in,\dot{X}}$ $\varphi_1(\dot{X}, v_0, v_1), \varphi_2(\dot{X}, v_0, v_1)$, *and* $y \in \mathbb{R}$, *so that*

$$\begin{aligned} \forall x \in \mathbb{R}(x \in A &\iff L_{\zeta^{B,y,x}}[B, y, x] \models \varphi_1[B, y, x] \\ &\iff L_{\zeta^{B,y,x}}[B, y, x] \models \neg\varphi_2[B, y, x]). \end{aligned}$$

(ii) $A \leq^\infty B$ *iff there are* Σ_2*-formulae in* $\mathcal{L}_{\in,\dot{X}}$ $\varphi_1(\dot{X}, v_0, v_1), \varphi_2(\dot{X}, v_0, v_1)$, *and* $y \in \mathbb{R}$, *so that*

$$\begin{aligned} \forall x \in \mathbb{R}(x \in A &\iff L_{\zeta^{B,y,x}}[B, y, x] \models \varphi_1[B, y, x] \\ &\iff L_{\zeta^{B,y,x}}[B, y, x] \models \neg\varphi_2[B, y, x]). \end{aligned}$$

The Lemma then identifies structures in which we can look to ascertain the outcomes of our ITTM computations relative to a set of reals B say. By way of analogy with ζ, the ordinal $\zeta^{B,y,x}$ is the least that is not ITTM-(B, x, y)-eventually-semi-decidable. It is thus also least such that $L_{\zeta^{B,y,x}}[B, y, x]$ has a proper Σ_2-elementary end-extension in the $L[B, y, x]$ hierarchy. If $\lambda^{B,y,x}$ is the least λ which is the height of a transitive elementary Σ_1-substructure of $L_{\zeta^{B,y,x}}[B, y, x]$, we could replace $\zeta^{B,y,x}$ in (i) with $\lambda^{B,y,x}$ of course.

3.2 Degree Theory and Post's Problem for ITTM-degrees

In [21] the degree theory of the ITTM-reducibility together with the jump operator $x \rightarrowtail x^\nabla$ is explored, pursuing the analogy with Turing degrees under Turing jump. We shall consider for the moment this simply on sets of integers. In [22] the same authors continue this investigation. In particular they consider the analogue of Post's

problem in both settings. With the definition of degree coming naturally from Def. 9(i) they establish that in fact there are no degrees at all between 0 and 0^{∇}.

In [77] degree theory was also explored but it pursued instead the analogy with hyperjump and hyperdegrees. Cor. 1 (i) establishes that the jump x^{∇} is essentially a mastercode, or equivalently at this level, the Σ_1-truth set, for a particular admissible set, namely that of $L_{\lambda^x}[x]$ where the natural association of $x \rightarrowtail \lambda^x$ satisfies the Spector criterion from Lemma 2(i). This is just as the hyperjump of x is the complete $\Sigma^1_1(x)$ set of integers, and is then recursively isomorphic to the Σ_1-Th$(L_{\omega_1^{x\,\mathrm{ck}}}[x])$. Just as there are no reals of hyperdegree between that of the hyperarithmetic sets and $\mathcal{O}$, we do not expect there to be any such intermediate degrees, which is just as the argument from [22] shows. As we iterate the jump operation (starting say from $0 = \varnothing$) we obtain iterates $0^{\nabla\alpha}$ in a linear fashion. At limit stages $\nu < \lambda$ we want to take $0^{\nabla\nu}$ as some minimally chosen upper bound $\{0^{\nabla\alpha} : \alpha < \nu\}$. For $\alpha < \zeta$ these are characterised as follows. We set $S = S^1_\zeta =_{\mathrm{df}} \{\mu < \zeta : L_\mu \prec_{\Sigma_1} L_\zeta\}$ the ordinals Σ_1-stable in ζ. By Theorem 3(iv) the least element of S is λ. Let μ_ι for $\iota < \zeta$ enumerate $S \cup \{0\}$. By standard arguments a non-projectible element of S must be a limit point μ_ν of S. (Note that easily L_ζ is a Σ_2-admissible set, and hence S has this maximal order-type.) Recall that ν is non-projectible if L_ν is a model of Σ_1-Separation. We let T^k_μ be the Σ_k-Th(L_μ). We already have seen that $0^{\nabla} \equiv_1 T^1_\lambda$ and $0^{\infty} \equiv_1 T^2_\zeta$. The following completes this picture in L below ζ.

Let $a_\alpha =_{df} T^1_{\mu_\alpha}$ unless μ_α is non-projectible, in which case let $a_\alpha =_{df} T^2_{\mu_\alpha}$. Define by recursion: $0^{\nabla 0} = \emptyset$; $0^{\nabla\alpha+1} = (0^{\nabla\alpha})^{\nabla}$; and $0^{\nabla\eta} = a_\eta$ for $Lim(\eta)$. Then we have:

Theorem 5 (Welch [77]) *(i) For $\alpha < \zeta$: (a) $\mu_{\alpha+1} = \lambda^{0^{\nabla\alpha}}$;*
(b) $0^{\nabla\alpha} \equiv_1 a_{\mu_\alpha}$ and
(c) $[0^{\nabla\alpha}]_\infty = \{z : z \in L_{\mu_{\alpha+1}} \backslash L_{\mu_\alpha}\}$. Hence it follows that:
(ii) $L_\zeta \models$ "The $=_\infty$-degrees are linearly ordered in type ζ and
$\mathrm{Lim}(\eta) \longrightarrow [0^{\nabla\eta}]_\infty$ *is the l.u.b. of $\{[0^{\nabla\alpha}]_\infty : \alpha < \eta\}$".*
(iii) $L_\Sigma \models$ "$\neg\exists y(\forall\alpha < \zeta)(0^{\nabla\alpha} <_\infty y <_\infty 0^\infty)$".

The fact that the jump may be iterated taking least upper bounds at limit stages *inside* L_ζ is no guarantee that l.u.b.'s exist outside of L and indeed it can be shown the natural hierarchy defined in this way stops at ω: there is no least upper bound of the $\{0^{\nabla n} : n < \omega\}$ in V but rather, continuum many minimal upper bounds (see [76]).

Q If $D = \{d_n : n < \omega\}$ is a countable set of $=_\infty$-degrees, does D have a minimal upper bound?

Although much is known and there are positive answers for Δ^1_{2n} degrees (assuming PD), for Δ^1_{2n+1}-degrees this remains open, even for hyperdegrees. *Minimal degrees* (compare minimal hyperdegrees) are known to exist by similar arguments for minimal hyperdegrees using Sacks style perfect set arguments (again see [76]).

If one turns to the other notion of jump $x \rightarrowtail x^\infty$ from Def. 1 (ii) the picture is somewhat similar, but there are significant differences: call an ordinal τ Σ_2-

extendible if there is $\sigma > \tau$ with $L_\tau \prec_{\Sigma_2} L_\sigma$. Let Z be the least Σ_2-extendible that is a limit of such. Then the natural hierarchy of $=^\infty$ degrees which takes l.u.b's at limit stages (where they exist) has length Z (rather than ω as for $=_\infty$-degrees).

Definition 12 *(i)* [21] *Let $A \subseteq 2^{\mathbb{N}}$, then* $A^\blacktriangledown =_{\rm df} \{(e,x) : P_e^A(x)\downarrow\}$.
(ii) $\mathbf{\Gamma}_0 =_{\rm df} \{A \subseteq 2^{\mathbb{N}} \mid A$ *is semi-decidable from a real* $\}$; *(iii)* $\mathbf{\Delta}_0 =_{\rm df} \mathbf{\Gamma}_0 \cap \neg\mathbf{\Gamma}_0$

Hamkins and Lewis in [21] showed that the decidable sets of reals, $\mathbf{\Delta}_0$, form a σ-algebra, closed under the Suslin $\mathcal{A}$-operator and properly containing the Selivanovski C-sets, and are within Δ^1_2. By a result of Solovay they are absolutely measurable, and have the property of Baire. Lemma 3(i) then describes abstractly the place of the class $\mathbf{\Gamma}_0$ within the Wadge hierarchy. The structure of sets of reals under either of the orderings $\leq_\infty$ or $\leq^\infty$ is dependent on one's set theory: if $V = L$ (or set generic extensions thereof) then following [32] there are $2^{\mathfrak{c}}$ mutually incomparable sets A, B below the complete Γ set. Following an argument of Steel [73], if we let Boolean($\mathbf{\Gamma}_0$) denote Boolean combinations of sets from $\mathbf{\Gamma}_0$, we have:

Theorem 6 Det(Boolean($\mathbf{\Gamma}_0$)) $\Longrightarrow$ *any complete $\mathbf{\Gamma}_0$-set is reducible to any $A \in \mathbf{\Gamma}_0 \backslash \mathbf{\Delta}_0$. Consequently there are no ∞-degrees between those of sets in $\mathbf{\Delta}_0$ and those of sets in $\mathbf{\Gamma}_0$.*

The solution to Post's problem then in this context depends on the ambient set theoretical universe. (See [79] for a further discussion of this, and [**?**] where it is shown that Det($\mathbf{\Gamma}_0$) proves the existence of inner models with a proper class of strong cardinals at the very least. However it is not known how sharp this bound is. One should contrast this strength with the corresponding result of Martin as it affects the Kleene degrees: Det(Boolean($\mathbf{\Pi}^1_1$)) - which implies there are no Kleene degrees between Borel and complete analytic, a result of Harrington [29] - is equivalent to $\forall x(x^\sharp$ exists$)$.) There are entirely similar results and effects for the eventually decidable reduction $\leq^\infty$. However if $\mathbf{\Gamma}$ is defined to be the class of sets eventually-semi-decidable in a real, then it is not known how to separate in terms of inner models Det($\mathbf{\Gamma}_0$) from Det($\mathbf{\Gamma}$).

3.3 Complexity of ITTM-Computation

Given the fruitfulness of complexity theory in the theory of (standard) Turing computation, it may seem also tempting to try and define complexity classes for their transfinite computations. In one sense Hamkins and Lewis initiated this with their attempts to pin down the clockable ordinals (think of 'time' here), and by showing, *eg*, that any arithmetic set could be decided in time less than ω^2 (Thm. 2.6 [21]). They had shown that no admissible ordinal is clockable: this can be shown directly as they do, and is in essence also an application of Π_2-reflection. Schindler [65] considered 'polynomial time'. He gave a more encompassing definition.

Definition 13 *(i) Let $A \subseteq 2^{\mathbb{N}}$, and $\alpha \leq \omega_1$, then we say that $A \in P^\alpha$ if there is $\beta < \alpha$ and an index e so that $\forall x \varphi_e(x)\downarrow^{<\beta}$ and $x \in A \leftrightarrow \varphi_e(x)\downarrow 1$.*

(ii) More generally if $\mathcal{D}$ is the class of (standard) Turing degrees and $f : \mathcal{D} \to \omega_1$ we say that $A \in P^f$, if there is an index e so that $\forall x \varphi_e(x)\downarrow^{<f([x]_T)}$ and $x \in A \leftrightarrow \varphi_e(x)\downarrow 1$.

We thus think of output 1 as 'acceptance'. Then (i) with $\alpha = \omega^\omega$ is Schindler's 'polynomial time' and was named P. He further observed that $P^{\omega_1^{\mathrm{ck}}} = \mathrm{HYP}$ and (both he and Hamkins) that P^{ω_1} coincided with those sets A that were $\Delta^1_1(\beta)$ for some $\beta < \omega_1$. Schindler defined $P^+ = P^{f_0}$ where $f_0(x) = \omega^x_{1\,\mathrm{ck}} + 1$. [14] (Thm 4) shows directly $P^{f_0} = P^{\omega_1^{\mathrm{ck}}}$. Underlying this is perhaps the following Bounding Lemma:

Lemma 4 (Bounding Lemma [80]) *(i) Let φ_e be a total computable function and suppose $\forall x(\varphi_e(x)\downarrow^{<\omega_1^x})$ then there is $\gamma < \omega_1^{ck}$ so that $\forall x(\varphi_e(x)\downarrow^{<\gamma})$.*

(ii) Let β be admissible, and φ_e a total computable function and suppose $\forall x(\varphi_e(x)\downarrow^{\leq\beta}$) then there is $\gamma < \beta$ so that $\forall x(\varphi_e(x)\downarrow^{<\gamma})$.

These are straightforward applications of, for (i), Σ^1_1-Bounding, and for (ii), Barwise Compactness. Schindler also defined some 'non-deterministic' classes:

Definition 14 *(i) Let $A \subseteq 2^{\mathbb{N}}$, and $\alpha \leq \omega_1$, then we say that $A \in NP^\alpha$ if there is $\beta < \alpha$ and an index e so that $\forall z \varphi_e(z)\downarrow^{<\beta}$ and $x \in A \leftrightarrow \exists y \varphi_e(x \oplus y)\downarrow 1$.*

(ii) More generally if $f : \mathcal{D} \to \omega_1$ we say that $A \in NP^f$, if there is an index e so that $\forall z \varphi_e(z)\downarrow^{<f([z]_T)}$ and $x \in A \leftrightarrow \exists y \varphi_e(x \oplus y)\downarrow 1$.

Then $NP = NP^{\omega^\omega}$. From the definition it is clear that $P \neq NP$ as the latter is clearly the class of Σ^1_1. Löwe defines various complexity classes and surveys their relationships in [51]. As he points out, Schindler does not explicitly give the connection between his NP classes and an actual non-deterministic ITTM. Löwe makes the appropriate definition and he suggests possible notions of *space* required for a computation $P_e(x)$; one such could be the supremum of the $L[x]$-constructible ordinal ranks of the snapshots of the scratch tape arising in the computation. We refer the reader to this paper for the discussion and question of the $PSPACE^f$ classes that arise.

[28] showed that for almost all f, $NP^f \neq P^f$ and later [14] showed that for many α $P^\alpha = NP^\alpha \cap \text{co-}NP^\alpha$ including those α that begin a gap in the clockable ordinals. (Such are admissible by [83] Thm. 50). However for many f, P^f is properly contained in $NP^f \cap \text{co-}NP^f$. If one restricts to sets of integers rather than sets of reals, a somewhat different picture emerges: for many β, for example β that are admissible and limits of gaps in the clockables then $P^\beta \cap \mathcal{P}(\mathbb{N}) = NP^\beta \cap \mathcal{P}(\mathbb{N})$ ([80]). For other β they are different.

It is arguably best to think of these results as epiphenomena on the constructible hierarchies that the ITTM's inhabit, with their relevant input; there is any case some

danger with this nomenclature that casual readers will be misled into thinking that these classes 'P^f','NP^f' *etc.* have something in common with the classical $P = NP$? problem, which really they do not.

3.4 Decidable *versus* eventually decidable

A feature that has not been much remarked on so far, is that there are two notions of ITTM-reducibility: that of semi-decidability and that of eventual semi-decidability, which, to take the simpler case of integers, are respectively decided by a Σ_1 relation over L_λ, or a Σ_2 relation over L_ζ. After all, even after all halting computations on integer input have halted the machine still carries on and may produce 'eventually' interesting data on the output tape. Furthermore the universal machine behaviour really does get its character from the Σ_2-extendible pair of ordinals (ζ, Σ). Is there then a case for singling out one of these reducibilities as somehow the 'primary' underlying relation? It is not part of the this survey's aims to examine applications of ITTM's, hence we shall not go in to the proposals for such applications that have been made for equivalence relation theory (such as in [8], [9], [10]) or in ITTM-computable model theory ([26]). Clearly when discussing such reducibilities, either a choice has to be made, or both reducibilities are carried along.

One might argue (as this author has on occasion) that in one sense the notion of eventual semi-decidability is the more general and encompassing, and more fully captures the essence of the machine action (whilst most of the papers cited tend to concentrate on the semi-decidable). That may seem satisfying, but on the other side there are arguments that the Spector pointclass of semi-decidable sets Γ_0 has better closure properties than that of the eventually semi-decidable Γ. For example, Theorem 1 of [18] shows that the Uniformization and Scale Properties hold for Γ_0 but both fail for Γ. This also for the generalisations of the ITTM's to those of Σ_n limit rules: there again Uniformization and Scale holds for the pointclasses Γ_n defined there for the Σ_1-semi-decidable notion. (It is open whether the weaker property holds that Γ relations can be uniformized by a Σ_3-function defined over L_Σ.) We leave this discussion at this point.

3.5 Correspondences with other theories

We discuss the relations between ITTM-reducibility and other theories. As is well known Π^1_1-monotone inductive definitions over, for example the structure $\mathbb{N}$ of the natural numbers (see, *e.g.*[64]), results in sets of integers that are themselves Π^1_1. It is also well known that strategies for Gale-Stewart games with open pay-off sets contained in $\mathbb{N}^\mathbb{N}$ are definable over $\langle L_{\omega_1^{\mathrm{ck}}}, \in\rangle$. We explore the connections and generalisations here.

Let $\Phi : \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})$ be any map, which in this context we call an *operator*. We call it 'recursive', 'arithmetical', 'Π^1_1', 'Γ', *etc.*, if the relation "$n \in \Phi(X)$" is recursive, ... Γ, and so on. It is *monotone* if $A \subseteq B \longrightarrow \Phi(A) \subseteq \Phi(B)$.

Definition 15 *Let* $\Phi : \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})$ *be a* Γ*-operator. We define the* Γ-quasi-inductive operator *using iterates of* Φ *as:*

$\Phi_0(X) = X; \qquad \Phi_{\alpha+1}(X) = \Phi(\Phi_\alpha(X));$

$\Phi_\lambda(X) = \liminf_{\alpha\to\lambda} \Phi_\alpha(X) =_{df} \cup_{\alpha<\lambda} \cap_{\lambda>\beta>\alpha} \Phi_\beta(X).$

We set the stability set *to be* $\Phi_{\mathrm{On}}(X)$.

It is easy to recast the ITTM computation steps on the tapes' contents as an example of a recursive quasi-inductive operator on $\mathbb{N}$, and in this case, *e.g.*, $\Phi_{\mathrm{On}}(\varnothing)$ for the universal machine is $\Phi_\zeta(\varnothing)$. We define $A \subseteq \omega$ to be a Γ-*quasi-inductive set* if there is a Γ-operator Φ with A (1-1) reducible to the stability set $\Phi_{\mathrm{On}}(\varnothing)$: $A \leq_1 \Phi_{\mathrm{On}}(\varnothing)$. (Again this should be compared to a set being *inductive* over $\mathbb{N}$ if it is (1-1) reducible to a fixed point of a Π^1_1-operator.) Recall the following characterisation of infinite two person games into sets $A \subseteq \mathbb{N}^{\mathbb{N}}$:

• Every Σ^0_1-game which is a winner for player I has a Π^1_1-monotone-inductive winning strategy (which in this context is equivalent to it being the stability set for a Π^1_1-monotone operator Φ; similar comments apply to the next bullet point).

• (Solovay - unpublished but see Kechris [34]) Every Σ^0_2-game which is a winner for player I has a Σ^1_1-monotone-inductive winning strategy.

One might tempted be to ask the question as to whether for Σ^0_3-games the winning strategies for I are ITTM-decidable, or putting it another way: recursively-quasi-inductive? It is easy to see that since an ITTM can emulate arithmetic operations, and even Π^1_1 ones, that any arithmetic or Π^1_1-quasi-inductive set is also recursively quasi-inductive. The answer turns out to be negative, but we shall see it is a close race.

Clearly a certain amount of analysis is needed to prove that there are wellorderings of sufficient length along which to run an ITTM and see that it either halts or loops. How much of second order number theory is needed for this? Burgess in [6] first defined, to our knowledge, *arithmetic-quasi inductive* (in a very different context) so in deference to that we formulate in second order number theory the existence of sufficiently long wellorderings for arithmetic operators, although by the comment in the last paragraph, this would have been equally valid using recursive, or Π^1_1-operators. By a 'repeat pair' of ordinals for a quasi-inductive operator Φ, we mean the least pair $(\zeta, \Sigma) = (\zeta(\Phi, x), \Sigma(\Phi, x))$ with $\Phi_\zeta(x) = \Phi_\Sigma(x) = \Phi_{\mathrm{On}}(x)$.

Definition 16 AQI *is the sentence: "For every arithmetic operator* Φ*, for every* $x \subseteq \mathbb{N}$*, there is a wellordering* W *with a repeat pair* $(\zeta(\Phi, x), \Sigma(\Phi, x))$ *in* $\mathrm{Field}(W)$*". If an arithmetic operator* Φ *acting on* x *has a repeat pair, we say that* Φ converges *(with input* x*).*

AQI thus asserts precisely that we have sufficiently long wellorders to discover the behaviour of quasi-inductions along them, or equivalently, the looping or otherwise behaviour of an ITTM machine. Typically then one may ask:

Q: *What is the strength of* $\mathsf{ACA}_0 + \mathsf{AQI}$?

We then have:

Theorem 7 (Welch)[84] *The theories:* $\mathsf{\Pi^1_3}$-$\mathsf{CA_0}$, $\mathsf{\Delta^1_3}$-$\mathsf{CA_0}$+ Σ^0_3-*Det,* $\mathsf{\Delta^1_3}$-$\mathsf{CA_0}$+AQI, *and* $\mathsf{\Delta^1_3}$-$\mathsf{CA_0}$ *are in strictly descending order of strength, meaning that each theory proves the existence of a* β*-model of the next.*

We remark here that in determining the proof-theoretic ordinal for $\mathsf{\Pi^1_2}$-$\mathsf{CA_0}$, Rathjen ([60]) was lead to analyse Σ_1-chains of arbitrary but finite length of models in the L-hierarchy of the form $L_{\gamma_0} \prec_{\Sigma_1} L_{\gamma_1} \prec_{\Sigma_1} \cdots \prec_{\Sigma_1} L_{\gamma_n}$. (Recall also that the least β-model of $\mathsf{\Pi^1_2}$-$\mathsf{CA_0}$ occurs as $\mathbb{R} \cap L_{\gamma_\omega}$ where γ_ω is the least γ so that L_γ is a union of an infinite such chain of Σ_1-substructures.) Speculatively, to analyse $\mathsf{\Pi^1_3}$-$\mathsf{CA_0}$ in the same manner will need a theory of such chains but with Σ_2-elementarity replacing Σ_1. Hence analysing the pair $L_\zeta \prec_{\Sigma_2} L_\Sigma$ derived from ITTM-theory is the first step along this way. The "closeness" of Σ^0_3-Det to AQI is indicated by the fact that a chain of length 2: $L_{\zeta_0} \prec_{\Sigma_2} L_{\zeta_1} \prec_{\Sigma_2} L_{\zeta_2}$ is sufficient to establish that $(\mathbf{\Sigma^0_3}\text{-Det})^{L_{\zeta_0}}$. (The fact of the matter is that the theories are even closer than just a chain of length 2, but part of the theorem above asserts that a chain of length 1 is insufficient - see [84] for a discussion of this.)

One conjecture we mention here since it concerns this level of determinacy, is a possible connection with R. Lubarsky's "*Feedback-ITTM's*": one envisages an ITTM which has the capability to call as a subroutine some other ITTM-computation for input, handing data over to the called machine and awaiting 'feedback'. This seems uncontroversial, but there is the possibility for infinite descending chains of calls. Lubarsky analyses the situation in [52]. One may prohibit descending chains by various devices: he does this by assigning ordinals to calls, and requiring naturally that further subcalls must attach lower ordinals. However he also discusses the situation where the machines are purposefully free to make such infinite chains of calls on certain inputs (such a computation he calls 'freezing'). The level of L where strategies for Σ^0_3-games first are definable, is precisely that β so that L_β has an illfounded end-extension $\mathcal{M}$ with infinite depth nesting of Σ_2 extendible pairs $L_{\gamma_i} \prec_{\Sigma_2} L_{b_i}$ with, for all $i < \omega$, $\gamma_i \leq \gamma_{i+1} <_{\mathcal{M}} b_{i+1} <_{\mathcal{M}} b_i$. Conjecturally there appears *prima facie* a connection between this level of determinacy and such a machine model.

Theories of Truth - a historical note

We remarked above that *arithmetically quasi-inductive* had been coined by Burgess. In [6] he analysed a *Herzberger Revision Sequence*. Herzberger ([30]) had defined in effect a revision operator just beyond arithmetic by defining, in a language for arithmetic with an additional predicate symbol $\dot{T}$ (for 'Truth'), for any $X = H_0 \subseteq \mathbb{N}$:

$H_{\alpha+1} = \{\ulcorner \sigma \urcorner : \langle \mathbb{N}, +, \times, \cdots, H_\alpha \rangle \models \sigma\}$; with H_α interpreting T;

$H_\lambda = \cup_{\alpha<\lambda} \cap_{\lambda>\beta>\alpha} H_\beta$.

Benedikt Löwe was the first to point out the similarity between this formalism of a revision sequence of truth sets and ITTM's, and in [50] also wrote a program that could produce any Herzberger sequence as long as it possessed ITTM computable

length. This left open the nature of the relationship between the two formalisms, but as may be gleaned from the above, essentially they are similar as in fact any H-revision sequence starting with $H_0 = x$ can be mimicked on an ITTM with similar input, irrespective of length, and indeed this is how one sees that essentially a 'repeat pair' for a Herzberger sequence must exist (as Herzberger had asserted). Burgess had calculated that repeat pair for $X = \varnothing$ as being precisely our (ζ, Σ) here and established these as least so that $L_\zeta \prec_{\Sigma_2} L_\Sigma$. This much earlier work on theories of truth was unknown to those working on ITTM's in 2000 when the significance of this pair of ordinals for ITTM theory was established in [77].

Burgess also then had defined arithmetical quasi-inductive sets as above. More recently Field (*cf* [16] and subsequently) has started to develop a theory of truth which uses revision theoretic models given by a Π^1_1-quasi-inductive operator. That all three formalisms are essentially mathematically similar - in that they all produce recursively isomorphic stable sets - is established in [82].

3.6 Varying the limit rules - hypermachines

One might ask whether other limit rules are conceivable. It was shown in [77] that the *Liminf* rule is in one sense 'complete' for Σ_2 limit rules. This is entirely unsurprising in view of the fact that the actions of any machine whose operations are recursive at successor steps, and further obeys a definable rule at limit stages that sets a cell's value according only to what has happened at previous stages, will be absolute to L: consequently one may argue that $C_i(\alpha)$ is always $\Sigma_2(L_\alpha)$ (assuming say integer input). Conversely given a wellorder w (either as input or self-computed) a subroutine may inductively define codes along w for L_β's for β less than the order type of w. The conclusion is that for many stages α a code for L_α will be uniformly arithmetic in the snapshot of a machine (and so in particular the universal machine) at precisely stage α. This is probably explicit and implicit in [21] and the earlier papers to varying degrees, but it is explicitly shown in [19] that one program computes on the one hand codes for J_α (levels of the Jensen hierarchy, for $\alpha < \Sigma$), whilst simultaneously on the other their Σ_2-truth sets, T^2_α say, (under some recursive enumeration of sentences). *Inter alia* a code for J_α is written by stage $\omega^2.(\alpha + 1)$. For limit stages μ this has to be done in such a fashion that the Σ_2-truth sets are (standardly) uniformly c.e. in the machine's snapshots at stages $\omega^2 \cdot \mu$. This uses the fact that for $\alpha < \beta_0$ (where $L_{\beta_0} = J_{\beta_0}$ is the least model of ZF^-) there are *uniform* Σ_n-skolem functions for each $n < \omega$. This requires one to show at limit stages μ that T^2_μ, although not necessarily equal to $\mathrm{Liminf}_{\beta\to\mu} T^2_\beta$, is nevertheless (standardly) uniformly c.e. in it.

Can we do this for more complicated machines with say a Σ_3 or a Σ_n 'limit rule' for cell values? This is investigated in [18] where such rules are supplied and an analysis of programs that, as above, produce again simultaneously codes for J_α and truth sets T^n_α for $\alpha < \Sigma(n)$, where now $\Sigma(n)$ is least so that $L_{\Sigma(n)}$ has a Σ_n-proper elementary substructure $L_{\zeta(n)}$. 'Σ_n-Machines' operating under such

limit rules exhibit entirely analogous behaviours as the ITTM's: a universal such machine will produce entries on its tapes until $\Sigma(n)$ but the snapshot at $\Sigma(n)$ will be equal to that at $\zeta(n)$, and thus it will enter a permanent loop of periodicity $\Sigma(n)$. The rules are defined inductively: those for $\Sigma(n)$ are defined assuming the rules for $\Sigma(n-1)$-machines. It is probably fair to say that as n increases the rules become increasingly distant from any machine intuition. Even for Σ_3 the limit rule is somewhat complicated, and depends on taking a liminf for a limit stage λ, not on all ordinals $\alpha < \lambda$ but roughly, as a first approximation, those of 'stable informational content'. More precisely, we might say that, relative to λ, an ordinal $\alpha < \lambda$ is of such stable informational content, if for any real $y \in 2^{\mathbb{N}}$ that has appeared on a tape before stage α, then if by stage λ something 'new about y has been written to the scratch tape' then actually that new fact had already been written at some stage less than α. The intuition is that at stage α, anything that is going to be known about such a y by stage λ is already known. One then determines cell values $C_i(\lambda)$ by taking a liminf of only those $C_i(\alpha)$ where the ordinals below α that are of stable content relative to α are those below α of stable content relative to λ.

The conclusion is that with more complicated limit rules and some effort, one can define machines that compute, for example, any real in the least β-model of analysis. Combined with recent work of Montalban and Shore [55] for example one should get:

Proposition 1 *Winning strategies for games in the k'th level of the difference hierarchy on (lightface) Σ^0_3 sets are computable by Σ_{k+1}-machines.*

3.7 Alternative ITTM machine models

One might ask: are three tapes necessary? what if one enlarges the alphabet? Enlarging the alphabet perhaps unsurprisingly does not change the class of functions that are computable, nor adding extra tapes. Perhaps more surprisingly reducing the number of tapes does change the class of computable functions. In [27] Hamkins and Seabold investigate what happens if the triple tape arrangement is reduced to a single tape of cells $\langle C_i \rangle_i$. Except for one small, but vital, difference, the class of computable functions remains the same: for $f : \mathbb{N} \to \mathbb{N}$, or $f : 2^{\mathbb{N}} \to \mathbb{N}$ there is absolutely no difference. However for $f : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ there is a difference. It is easy to arrange a pre-run of the program that expands the input so that it lies only on the cells $\langle C_{3i+1} \rangle_i$ now, with 0's on the other cells. One then may successfully mimick the run of a 3-tape-ITTM machine on the single tape using $\langle C_{3i} \rangle_i$ and $\langle C_{3i+2} \rangle_i$ as the two other 'tapes' for scratch work and output. If this successfully comes to a conclusion there is the natural output y written on $\langle C_{3i+2} \rangle_i$, and we need to squash this back down on to the whole tape $\langle C_i \rangle_i$ erasing the other values. How does one know when this has been completed? In [27] it is shown that this problem cannot be surmounted. Different solutions can be proposed: a special tab cell, not on the tape $\langle C_i \rangle_i$ and reserved precisely for this purpose is one. Another is to allow a special symbol $*$ say onto the tape to mark the progress of this squashing process.

One solution advocated in [79] is that of allowing the alphabet of a 1-tape machine to consist of $0, 1$ and 'B' the latter for a blank or empty cell. One then changes the limit rule so that if at limit stage μ C_i has changed value cofinally often in μ, then $C_i(\mu)$ is set to B - thus B represents ambiguity or non-determination to some extent. (In the other cases the cell values are set to that fixed value on a tail below μ of course.) This then remarkably also suffices to compute the full complement of 3-tape-ITTM-computable functions $f : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$.

4 Longer tapes

Once one has freed Turing's machine from the confines of time, it is tempting to free up some extra space too. One can allow for cell sequences of arbitrary ordinal length and even consider a machine acting on a proper class sequence of cells C_i for each ordinal $i \in \mathrm{On}$. The idea is that one keeps the programs fixed and the $\liminf$ limit rule as for an ITTM. We must devise some specific rule to handle the limit cells $C_\omega, C_{\omega+\omega}, \ldots, C_\lambda, \ldots$ when the R/W head is over a limit cell but then receives an instruction to move left. One formalism is to just to return the head to the start of the tape C_0. The position function for the head at time α, $l(\alpha)$, is then governed by the $\liminf$ rule, and thus can now ascend to higher limit level cells.

4.1 Ordinal length tapes

What can such machines now compute? Essentially we are allowing an infinite sequence of $0, 1$'s to come into play and there is the possibility of coding arbitrary sets by such sequences. Dawson and Koepke came up independently with this concept. Dawson in [13] formulated an *Axiom of Computability* that said that every set could appear coded on the output tape of such a machine whilst it was running; thus at some point for some program P_e a machine (not necessarily halting) would have a code for a set y on the output tape for any set y. He then proves that the class of such sets form a transitive class satisfying ZFC. Studying the grid of snapshots produced yields a form of Löwenheim-Skolem theorem that allows for a proof of the Condensation Lemma and so the Generalised Continuum Hypothesis. The "computable sets" of this construction are of course the same as the constructible sets of the Gödel hierarchy, since as remarked earlier the machine theoretic operations are absolute to L. Koepke in [41] and with Koerwien in [42] considered halting computations starting from an On-length tape with input only finitely many 1's representing a finite set of ordinal parameters. Remarkably one has:

Theorem 8 (Koepke [41]) *A set $x \subseteq \mathrm{On}$ is* On-ITTM*-computable from a finite set of ordinal parameters if and only if it is a member of the constructible hierarchy.*

The left to right direction is just the observation that the machine operations are absolute to L. It is the converse that takes some work. Koepke goes into considerable

programming detail to show that the *bounded truth predicate*:

$$\{(\alpha, \varphi, \vec{x}) \mid \varphi \text{ an } \in\text{-formula}, \alpha \in \mathrm{On}, \vec{x} \in L_\alpha, L_\alpha \models \varphi(\vec{x})\}$$

is computable. The reduction of sets in L to finite sequences of ordinals is achieved by noting (by an induction on α) that any set $x \in L_\alpha$ can be defined by one of countably many terms of $\mathcal{L}_\in$, evaluated over L_α, whilst using lower levels as parameters: $t(L_{\alpha_0}, L_{\alpha_1}, \ldots, L_{\alpha_k})$. This allows a bounded truth function of the form: $W(a, \varphi) = 1 \leftrightarrow \varphi(a)$ to be recursively defined for bounded formulae φ. Once the theorem is proven, then GCH and $\diamondsuit$ arguments follow by considering Löwenheim-Skolem arguments applied to computations defined inside transitive ZF^- models.

An unrealised hope was that computational models, which slow down the construction of the L hierarchy, might, rather like the *Silver machine* precursors for constructing L, yield up alternatives to, or at least insights into the more delicate fine structural arguments that go into proving $\square$ or morasses. Silver machines were indeed invented to avoid the Jensen hierarchy of J_α's for this very purpose (see [4] and [15]). Both the Silver machines, and the *Hyperfine Structure* of Friedman and Koepke ([17]) however make use of a Finiteness Property that works against the ITTM formalism (as indeed does the Σ_2-nature of the limit rules.)

4.2 Admissible length tapes

Another natural candidate is to compare computable reducibilities of a machine model with length of tape an admissible ordinal $\alpha > \omega$ with α-recursion theory.

Takeuti [74] seems to have been the first to consider (at least in print) computation involving ordinals themselves. He considered a scheme equivalent to Σ_1-definability on ordinals to replace the notion of 'recursive enumerability'. There were a number of other attempts to extrapolate from Kleene's equational calculus to ordinal valued functions. Such calculi had been developed by Machover [53], Levy [49], Tugué [75], Kripke [48], Platek [58]. Kripke-Platek set theory emphasised the utility of Σ_1-Replacement and the notion of an *admissible set* and concomitant weakening of ZF as an axiom system emerged.

Although some of the above schemes involved forms of an equational calculus (such as Kleene's) and others imagined ordinals *per se* as the objects of computation (see, *e.g.* , the discussion in [64] Part V for an account of some of this development), we shall first examine ITTM's with lengths of tape an arbitrary admissible ordinal, and in a later section consider machine models dealing with ordinals as objects directly. Clearly an α-ITTM with tape an admissible α is just a cut-down version of the On-ITTM just described. The sets imagined as being described by such a version of computation would be those that could be enumerated, or traversed as input, in α many steps.

In terms of α-recursion theory a set $A \subseteq \alpha$ which is $\Sigma_1(L_\alpha)$ is called *α-recursively enumerable* (α-r.e.). It is *α-recursive* if both it and its complement are α-r.e. and thus it is $\Delta_1(L_\alpha)$. For the notion of relative α-recursion a notion of reduction

procedure was defined; originally a notion now called *weak α-recursive* $A \leq_{w\alpha} B$ was employed, but this was discovered by Driscoll [33] to be intransitive. The notion of $A \leq_\alpha B$, A is *α-recursive in B* is more stringent and $\leq_\alpha \subsetneq \leq_{w\alpha}$ in general. Koepke and Seyfferth in [44] use the restriction of Koepke's ordinal tape machines to an α, and define *A is computable in B* to mean that the characteristic function of A can be computed by a machine in α many stages from an oracle for B. For them this is exactly the relation that $A \in \Delta_1(L_\alpha[B])$. This is again a strictly proper weakening of the relation $\leq_{w\alpha}$. However this has the advantage that the notion of α-computability and α-computable enumerability tie up exactly with the notions of α-recursiveness and α-r.e. With this modelling of α-r.e they proceed to give a reworking of the argument of Sacks-Simpson Theorem ([62]) for solving Post's problem:

Theorem 9 (Sacks-Simpson [62]) *There are α-computable enumerable sets A, B with both $A \not\leq_{w\alpha} B$ and $B \not\leq_{w\alpha} A$*

This approach although it ties up neatly with the notions of α-recursion, and is a very natural extension of the Turing model to α-length tapes, has again the disadvantage that the relation of 'computable in' is intransitive.

Dawson addressed this problem in [13]. He considered various modes of behaviour: a 'punch-hole tape' of length α that can only be written to once; and '*α-sequential computation*' whereby the cells of the output tape are written out in sequential order. The term 'punch-hole' refers to the fact that the output tape may be written to once only. (Such ω-tape versions were also initially considered by Hamkins and Kidder as prototypes for an ITTM. It is an easy observation (S-D Friedman and the author) to check that such ω-tape machines decide all and only arithmetic predicates and, when halting on integer input, do so by ω^2.) The effect of the restriction to α-sequential computation is that any α-length computation has all the initial segments of its output as elements of L_α. In terms of traditional α-recursion theory we should say that the initial segments of the computation are thus *α-finite*. An α-r.e. set with all initial segments α-finite was called *regular*. This is in contradistinction to a general α-recusively enumerable set (for example there are ω_1^{ck}-r.e. subsets of ω that are not in $L_{\omega_1^{\mathrm{ck}}}$). α-recursion theorists recognized the utility of regular α-r.e. sets: proofs are more tractable when sets can be assumed to be regular. Usefully it was shown by Sacks ([63]) that any α-degree of α.-r.e. sets contains a regular set. This points the way to attempting to find a notion of computation which restricts attention to regular sets.

Dawson defines:

Definition 17 *(i) A is uniformly-α-computably-enumerable in B ("A is α-c.e. in B") if there is a program index e and ordinal parameter $\eta < \alpha$ for which, when given B as input on one tape, there is a function $\delta : \alpha \to \alpha$ so that it will write to the punch-hole output tape in such a way that $A \restriction \gamma$ will be written by time $\delta(\gamma)$.*

(ii) A is uniformly-α-computable in B ($A \leq_c B$) if both A and cA are α-c.e. in B.

The presence of the function δ thus simply enforces regularity: $A \cap \gamma \in L_\alpha[B]$ for any $\gamma < \alpha$. He then has:

Theorem 10 (Dawson) *A is α-c.e if and only if it is α-r.e. and regular. A is α-computable if and only if it is α-recursive.*

Although Dawson does not discuss the other useful notion for classical α-recursion theory, *hyperregularity*, it is easy to argue that an α-c.e set A is hyperregular if there is pair (e, η) and an *α-computable* function δ all as in Definition 17(i) outputting A.

The upshot of Dawson's definition is that he is considering α-degrees, but restricting to where all α-c.e. sets are α-regular, and any set that is A-α-c.e. will be A-α-regular *etc.* He then has that the structure of the α-degrees of the α-r.e. sets in the classical sense, is isomorphic to that of the α-computable-degrees of the α-c.e. sets. Hence those theorems of the classical α-recursion theory about α-r.e. sets whose proofs rely on, or use regular α-r.e. sets will carry over to his theory. These are such as the Sacks-Simpson theorem above: there are α-c.e. sets A, B that are α-computably, $\leq_c$-incomparable. However the Shore Splitting Theorem (that any regulat α-r.e. sets A may be split into two disjoint α-r.e. sets B_0, B_1 with $A \not\leq_\alpha B_i$, [68]) is less amenable to this argument as in general taking unions and intersections does not preserve the isomorphism between the α-r.e. degrees and the α-c.e. degrees.

The proof of the Shore Density theorem (that between any two α-r.e. sets $A <_\alpha B$ there lies a third α-r.e. C: $A <_\alpha C <_\alpha B$, [69]) is more complex than the other arguments, and seemingly relies on more of the fine structure of L for its proof. This can however be generalised, and for predicates $\mathbb{B}$ that result in so-called *acceptable* and *sound* hierarchies (see, *e.g.*, [66] Defs.1.20 & 5.7,) Dawson lifts the notion of α-computation to $\mathbb{B}$-α-computation where now $\underline{\mathbb{B}} =_{\rm df} \langle J_\alpha^{\mathbb{B}}, \in, \mathbb{B}\rangle$ is a transitive, admissible, acceptable, and sound structure. These assumptions make the structure $\underline{\mathbb{B}}$ sufficiently L-like for Shore's arguments to go through. He then has (working just in the case that $\mathbb{B} \subseteq \alpha$):

Theorem 11 (Dawson - The α-c.e. Density Theorem) *Let $\underline{\mathbb{B}}$ be as above. Let A, B be two $\mathbb{B}$-α-c.e. sets, with $A <_{\mathbb{B},\alpha} B$. Then there is C also $\mathbb{B}$-α-c.e, with $A <_{\mathbb{B},\alpha} C <_{\mathbb{B},\alpha} B$.*

5 Infinite Time Register Machines

5.1 Number register machines

Koepke has been instrumental in looking at the behaviour of *register machines* such as those of Shepherdson and Sturgis [67], as described in Cutland [11], or those of or Minsky [54]. Such machines have a finite number of natural number registers R_i for $i < N$, acting under a program consisting of a finite list of instructions

$\vec{I} = I_0, \ldots, I_q$. These instructions come from a standard set for such machines of incrementing or setting to zero, a register, transferring of register contents and a conditional jump on comparison of two registers to another instruction. We shall say that the machine at time τ is about to perform instruction $I(\tau)$, and the register contents are $R_i(\tau) \in \mathbb{N}$ for $i < N$. We let $\vec{R}(\tau)$ be the N-vector of the registers' contents at time τ. We specify that at limit times λ the next instruction to be performed $I(\lambda) =_{\mathrm{df}} \liminf_{\alpha \longrightarrow \lambda} I(\alpha)$. The reader may like to verify that result of this is that the machine is at the beginning of the outermost subroutine in the program that was called unboundedly often before time λ; register values at limit stages are also defined according to $\liminf^*$ values:

$$R_i(\lambda) =_{\mathrm{df}} \liminf_{\alpha \to \lambda} R_i(\alpha) \textit{ if this is finite; otherwise we set } R_i(\lambda) = 0.$$

It is this resetting of unbounded values to zero (rather than allowing the machine to crash) that gives the model its surprising strength. A function $F : \mathbb{N}^N \to \mathbb{N}$ is then *ITRM-computable* if there is some program P of the above sort with $P(\vec{k}) \downarrow F(\vec{k})$ for every $\vec{k} \in \mathbb{N}^N$. In order to handle reals, or sets of integers, we allow the machine to have an oracle query as instruction. Thus a register R_i may be reset to zero if its contents is in an oracle set $Z \subseteq \mathbb{N}$. This machine was investigated in [7], [43].

In the former paper the clockable ordinals are again defined as the lengths of times taken for a halting computation, and in general the relations between clockable ordinals and computable ordinals are investigated in the same spirit as Hamkins and Lewis did for ITTM's. Here however the clockables form an initial segment of the countable ordinals, and every *computable ordinal* (that is an ordinal which has real code whose characteristic function is ITRM-computable) is clockable. They show that the ITRM's are (perhaps unsurprisingly) weaker than the ITTM's as the latter can simulate the former and calculate their halting sets.

Definition 18 (n-register halting set)

$$H_N =_{\mathrm{df}} \{\langle e, r_0, \ldots, r_{N-1}\rangle \mid P_e(r_0, \ldots, r_{N-1}) \downarrow\}.$$

They give a criterion for when a machine has started looping after θ many steps. Note that the two functions I and $\vec{R}$ give the course of computation for θ steps.

Lemma 5 *Let $I : \theta \longrightarrow \omega, R : \theta \longrightarrow {}^N\omega$ be a computation of the N register machine with program P and with oracle Z for order type θ many stages. Then if this computation has not halted by stage θ, then it will never do so if θ is sufficiently large so that there is some constellation (I', R') so that*

$$\mathrm{otp}(\{\beta \mid I(\beta) = I' \wedge R(\beta) = R'\} \geq \omega^\omega.$$

They present an algorithm using a stack that when programmed on such a machine gives a yes/no output as to whether the oracle Z contains a set of integers

coding a wellfounded relation. This is a back-tracking algortihm that searches for the leftmost descending path through Z's coded ordering, if it exists. Thus Π^1_1-complete sets are decidable by such ITRM's. It is also easy to see that Boolean combinations of Π^1_1 sets are also decidable. They also prove

Theorem 12 (Koepke-Miller [43]) *For any N the N-halting problem: '$\langle e, \vec{r}\rangle \in H_N$' is decidable by an ITRM. Similarly for any oracle Z, the (N, Z)*-halting problem *'$\langle e, \vec{r}\rangle \in H^Z_N$' is decidable by a Z-ITRM with an oracle for Z.*

Crucially in the above as N increases the number of registers needed to run a program deciding H_N also increases. As they remark this shows (in contradistinction to ITTM's, or indeed finitary register machines) that there can be no universal ITRM.

In order to ascertain the exact strength of ITRM's it is argued in [47] that a simple one register machine (!) must halt or enter an infinite loop by the *second* admissible ordinal. Note that $\omega_1^{\rm ck}$ is not *a priori* a candidate here: if $\mathrm{Liminf}_{\beta\to\omega_1} R_0(\beta) = p < \omega$ then indeed a Π_2-reflection argument shows that on a closed and unbounded in $\omega_1^{\rm ck}$ set of ordinals, the same instruction number and this liminf value of p are revisited, and so by the Lemma 5 the machine is looping; however there is the possibility that this liminf value is ω for the first time and then $R_0(\omega_1^{\rm ck})$ is reset to 0. At $\omega_1^{\rm ck} + \omega_1^{\rm ck}$ this could reoccur, but now the next instruction number could differ! However this pattern cannot go on for ever and by $\omega_2^{\rm ck}$ the machine must halt or loop. An induction on N shows that each time a register is added another admissible ordinal is needed for this argument. Hence by adding registers we can find programs that clock any ordinal below $\omega_\omega^{\rm ck}$ - the first limit of admissibles. The argument above can be used to show that given sufficient admissible ordinals we can prove that the halting sets H_n exist. More formally in the case of number register machines, a well known subsystem of second order number theory measures their strength.

Theorem 13 (Koepke-Welch [47])
(i) Let $\mathsf{ITRM_N}$ *be the assertion: "The* N-register halting set H_N *exists."*
Then: $\mathsf{KP}+$*"there exist* $N+1$ *admissible ordinals* $> \omega$*"* $\vdash \mathsf{ITRM}_N$.
(ii) Let ITRM *be the similar relativized statement that "For any* $Z \subseteq \omega$*, for any* $N < \omega$ *the* N-register halting set H^Z_N *exists." Then:* $\mathsf{\Pi^1_1\text{-}CA_0} \vdash \mathsf{ITRM}$.

Now for the converse (where $\mathrm{HJ}(N, y)$ denotes the n'th hyperjump of y):

Theorem 14 (Koepke-Welch [47]) *(i)* $\mathsf{ATR_0} + \mathsf{ITRM} \vdash \mathsf{\Pi^1_1\text{-}CA_0}$.
(ii) There is a fixed $k < \omega$ so that for any $N < \omega$
$\mathsf{ATR_0} + \mathsf{ITRM_{N\cdot k}} \vdash$ "$\mathrm{HJ}(N, \varnothing)$ *exists.*".

5.2 Ordinal Register Machines (ORM)

Koepke also considered register machines that contained ordinals. Platek (in private correspondence) indicated that he had thought of his equational calculus on

recursive ordinals as being also implementable as some kind of ordinal register machine. Ryan Siders had also been considering such machines and in a series of papers they considered what could be done with unbounded ordinals. The format is the same as for ITRM's except now that we allow ordinals as register entries; there is no longer any need for a register's contents to be reset to 0 if a liminf becomes unbounded in ω (or any larger limit ordinal λ) at a limit stage. Then the ordinal arithmetic operations of addition, mulitplication and exponentiation can be shown to be ORM-computable, as well as the Gödel pairing function.

Theorem 15 (Koepke-Siders [46]) *A set $x \subseteq \mathrm{On}$ is* ORM*-computable from a finite set of ordinals parameters if and only if it is a member of the constructible hierarchy.*

They do this as indicated for the OTM model in Sect 4.1 above by following that plan: as G and the ordinal arithmetic is ORM computable, they may show that the recursively given truth predicate $T \subseteq \mathrm{On}$ is ORM-computable. They thus can characterize ordinal computability by the theorem above.

They remark that their program is implementable on a 12 register machine, and assert that this can even be lowered to 4 - so conceptually this is a rather simple machine.

Hamkins and Miller at roughly the same time took Koepke and Siders ORM-model and proved a number of facts about them. First a definition of jump.

Definition 19 *Let P_e be the e'th ORM program. (i) The* (weak) jump *is the set*
$$0^{\Diamond} = \{e \in \mathbb{N} \mid P_e(0)\downarrow\} \ ;$$
(ii) The strong jump *is the set*
$$0^{\blacklozenge} = \{(e, \alpha) \in \mathbb{N} \times \mathrm{On} \mid P_e(\alpha)\downarrow\}$$

For finite time register machines the sets of which these are analogues are recursively isomorphic, but in the infinite case they differ. Indeed if we let $<_{\mathrm{ORM}}$ be the natural relative computability relation, we have $0 <_{\mathrm{ORM}} 0^{\Diamond} <_{\mathrm{ORM}} 0^{\blacklozenge}$. Hamkins and Miller show:

Theorem 16 (Hamkins-Miller [25]) *Let $F : \mathrm{On}^n \to \mathrm{On}$ be any partial function. Then F is* ORM*-computable if and only if it is* OTM*-computable.*

They further show:

Theorem 17 (Hamkins-Miller [25] *as abstracted in* [24]) *There exist* ORM*-computable enumerable sets A, B sets of ordinals both below $0^{\Diamond}$ which are $\leq_{\mathrm{ORM}}$-incomparable.*

They give an explicit proof of this fact, with a Friedberg-Muchnik style priority argument. As they also remark, a softer proof of this fact can be deduced from the observation due to Koepke, that, defining γ-ORM machines where ordinals parameters are restricted to γ, and run-times are less than γ, then the γ-ORM-computable

sets coincide with the γ-recursive sets (in the sense of α-recursion theory). Here γ is the supremum of the clockable ordinals (in the ORM-sense, which they remark in this case are easily argued to be all ORM-writable). Hence the result actually follows by the Sacks-Simpson Theorem 9 (and the proof could indeed be translated over). They also prove that A, B must be unbounded subsets of γ: shorter sets will not work.

In [23] the authors take up the observation that an ITTM can essentially in ω-steps decide Σ^0_2 truths (note again the Σ_2-nature of the limit rule) and by repeating this can decide $\Sigma^0_{2.n}$ truth within $\omega \cdot n$ steps. However an ORM needs to take times unboundedly times in ω^ω to decide arithmetic truths. Both styles of machine catch up with each other at stage ω_1^{ck}, with membership in hyperarithmetic sets of integers being so decidable.

6 Infinite Time Blum-Shub-Smale Machines and polynomial time set recursion

It is natural to consider what other models of computation might be capable of running transfinitely. The computational model ([5]) of Blum-Shub-Smale is a natural candidate. Such machines act on the real continuum directly rather than on Cantor or Baire space. A study of an infinite time version has been initiated by Koepke and Seyfferth ([45]). Such machines can be thought of as having a finite number of registers $R_1, \cdots, R_N$ containing reals $r_1, \ldots, r_n$. A finite program, which can best be thought of as a flow diagram with either conditional branching nodes, or function nodes where a rational function computation takes place. (A test is made to avoid division by zero.) At limit stages of time λ the current instruction number is set to be the $Liminf$ of the previous instructions, but the register values $R_i(\lambda)$ must be the continuous limit of the values $R_i(\alpha)$ for $\alpha < \lambda$. If for any register R_i, the previous values in it fail to converge to a unique limit then the whole computation is deemed to crash. This requirement of continuity on real values is quite stringent and as we shall see limits the machine's capabilities. However, and usefully, such computations as $sin(x)$, $e^x \ldots$ *etc.* can now be calculated with the machine halting after ω steps with the correct values (they cannot on the finite time model).

The requirement on continuity at limit points of time force some coding tricks on the programmer in order to get the machine to recognise limit ordinals: typically if one real register is thought of as having the function of a "scratch tape" then at limit stages one does not normally expect a continuous limit. However by continually dividing by 2, at a limit stage the tape will have the real 0.0. If a register is in danger of becoming unboundedly large at a limit time then it is best to calculate $\frac{1}{x}$ than x. With such devices as these, together with coding ω sequences of $0, 1$ bits into decimals, the authors of [45] can ensure that a machine with n nodes in its flow diagram, can halt (on rational number input) at ordinal times up to ω^{n+1}, but this

is the correct upper bound. Hence such machines, as a class, will on rational input either crash, halt, or be in an infinite loop by stage ω^ω.

As they ask, the question arises as to what is the computational scope of such machines. One observation is that the construction of such machines, as always, is absolute to the constructible L-hierarchy. Hence methods we have discussed above ensure that imagining the machine running on rational input with a final output can be done inside L_{ω^ω}. Hence the machine can only produce reals constructible by exactly this level. However they ask if it can compute all reals in L_{ω^ω}? We may answer this as follows: a program can be written to compute successive iterates of the following continuously decreasing hierarchy, we set:

$$h_0 = \mathbb{N}; \quad h_{\alpha+1} = (h_\alpha)' \cap h_\alpha; \quad Lim(\lambda) \to h_\lambda = \cap_{\alpha<\lambda} h_\alpha.$$

(Here $(h_\alpha)'$ denotes the standard Turing jump.) By a result of Shoenfield this hierarchy continues throughout the recursive ordinals, and every hyperarithmetic is (ordinary Turing) reducible to a member of this sequence. We can restrict this to just ω^ω stages and have an ITBSS-machine compute a code for any element of this initial segment. Hence we have an exact characterisation of the ITBSS computable reals as those recursive in some h_α for an $\alpha < \omega^\omega$. One may then argue that these coincide with the reals of L_{ω^ω}.

One might consider such machines with the ability to take $Liminf$'s at limit stages rather than just continuous limits; however now the notion of $Liminf$ is in the topology of the reals. An upper bound for the reals such a model could produce is that of the reals in $L_{\pi(3)}$ where $\pi(3)$ is the least Π_3-*reflecting ordinal*. However it is not known if this the best possible.

Finally we mention another connection of the ω^ω'th level of constructibility with the *Safe Recursive Set Functions* of Beckmann, Buss and S. Friedman ([2]). They develop a notion of set recursion using the notion of safe recursion developed on integers by Bellantoni and Cook. The key idea is to divide variables in a many-place function into *safe* and *normal*: such as $f(\vec{a}/\vec{b})$. Recursion is only allowed on safe variables - here $\vec{b}$. However input may now be sets, whether integers, ordinals, strings or otherwise. This typing of the variables ensures that ranks of sets are kept low by any $SRSF$-function, indeed they prove the following:

Theorem 18 *Let f be any $SRSF$. Then there is a ordinal polynomial q_f so that*

$$rk(f(\vec{a}/\vec{b})) \leq \max_i rk(a_i) + q_f(rk(\vec{a})).$$

Using an adaptation of Arai, such functions, whilst restricting to finite strings, may decide problems computed by (standard) polynomial time Turing machines. On the other hand it is shown that the least $SRSF$-closed set which contains ω as an element is again L_{ω^ω} which arose as above for $ITBSS$-machines. Hence the $SRSF$ functions computing on 2^ω to 2^ω will compute the same functions as an $ITTM$ machine bounded by that ordinal number of stages. This coincides with

Schindler's definition of 'polynomial time' functions for $ITTM$'s mentioned above. Thus we have three different notions of polynomial time for sequences on ω (or reals) from three different computational or recursion notions coinciding. This leads dangerously close to formulating a thesis that the continuous limit ITBSS machines (or is it the ITTM's up to stage ω^ω?) capture the informal notion of 'polynomial time infinitary computation'.

7 Conclusions

On reflection, what the above all illustrate is that there were always undiscovered avenues of concrete machine models that can contribute to our knowledge or assessment of what is 'generalised computation'. By way of being wonderfully simple examples of higher type recursion they throw a light on the earlier abstract (and at times difficult) theory. The set theorist might conclude that what the models do (at least in their oracle free versions) is mostly construct sets at the bottom of the Gödel constructible hierarchy L. The examples of Koepke and his co-workers of ITTM's and ITRM's, show that we can give a presentation of L very much in a computational spirit, and that presentation is yet another to lay besides the Gödel, Jensen, Silver, and the Friedman-Koepke hyperfine structural versions. The computability theorist may have a different view of matters and will see that there is a wealth of inventiveness when it comes to generalising the actual model constructs to allow for transfinite resources, and may calculate precisely how rapidly the computational strength of such machines grows. Then the reverse mathematician can gauge this strength in terms of axiomatic theories. Thus the theory of such machine models constitutes a very pleasing nexus between various branches of logic and computability theory.

Acknowledgements: The author would like to express his thanks to many people for their time and energy on this subject, in particular to Peter Koepke, and to Benedikt Löwe for, *inter alia*, pointing out to me the formal similarity between ITTM's and the Herzberger version of Revision Theory, but most particularly to Joel Hamkins who introduced me to the delightful ITTM's whilst in Kobe.

References

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

[2] A. Bechmann, S. Buss, and S-D. Friedman. Safe recursive set functions. *Centre de Recerca Matematica Document Series, Barcelona*, 2012.

[3] E. Beggs and J.V. Tucker. Can newtonian systems, bounded in space, time, mass and energy compute all functions? *Theoretical Computer Science*, 371:4–19, 2007.

[4] A. Beller and A. Litman. A strengthening of Jensen's $\square$ principles. *Journal of Symbolic Logic*, 45(2):251–264, 1980.

[5] L. Blum, M. Shub, and S. Smale. On a theory of computation and complexity over the real numbers. *Notices of the American Mathematics Society (N.S.)*, 21(1):1–46, 1989.

[6] J.P. Burgess. The truth is never simple. *Journal of Symbolic Logic*, 51(3):663–681, 1986.

[7] M. Carl, T. Fischbach, P. Koepke, R. Miller, M. Nasfi, and G. Weckbecker. The basic theory of infinite time register machines. *Archive for Mathematical Logic*, 49(2):249–273, 2010.

[8] S. Coskey. Infinite time Turing machines and Borel reducibility. In *Mathematical Theory and Computational Practice*, volume 5635 of *Lecture Notes in Computer Science*, page 129133. Springer, 2009.

[9] S. Coskey and J.D. Hamkins. Infinite time decidable equivalence relation theory. *Notre Dame Journal for Formal Logic*, 52(2):203–228, 2011.

[10] S. Coskey and J.D. Hamkins. Infinite time turing machines and an application to the hierarchy of equivalence relations on the reals. In *Effective Mathematics of the Uncountable*, to appear.

[11] N. Cutland. *Computability: an Introduction to Recursive Function Theory*. CUP, 1980.

[12] E.B. Davies. Building infinite machines. *British J. for Philosophy of Science*, 52(4):671–682, 2001.

[13] B. Dawson. *Ordinal time Turing computation*. PhD thesis, Bristol, 2009.

[14] V. Deolalikar, J.D. Hamkins, and R-D. Schindler. $P \neq NP \cap co\text{-}NP$ for the infinite time Turing machines. *Journal of Logic and Computation*, 15:577–592, October 2005.

[15] K. Devlin. *Constructibility*. Perspectives in Mathematical Logic. Springer Verlag, Berlin, Heidelberg, 1984.

[16] H. Field. A revenge-immune solution to the semantic paradoxes. *Journal of Philosophical Logic*, 32(3):139–177, April 2003.

[17] S-D. Friedman and P. Koepke. An elementary approach to the fine structure of L. *Bull. Symb. Logic*, 3, no. 4:453–468, 1997.

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

[19] S-D. Friedman and P.D. Welch. Two observations concerning infinite time Turing machines. In I. Dimitriou, editor, *BIWOC 2007 Report*, pages 44–47, Bonn, January 2007. Hausdorff Centre for Mathematics. Also at http://www.logic.univie.ac.at/sdf/papers/joint.philip.ps.

[20] R.O. Gandy. On a proof of Mostowski's conjecture. *Bulletin de l'Academie Polonaise des Sciences (série des sciences mathématique, astronomique et physique)*, 8:571–575, 1960.

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

[22] J.D. Hamkins and A. Lewis. Post's problem for supertasks has both positive and negative solutions. *Archive for Mathematical Logic*, 41:507–523, 2002.

[23] J.D. Hamkins, D. Linetsky, and R. Miller. The complexity of quickly ORM-decidable sets. In S.B. Cooper and A. Sorbi, editors, *Computation and Logic in the Real World: Proc. of CiE2007, Siena,*, volume 4497 of *Lecture Notes in Computer Science*, pages 488–496, Berlin, 2007. Springer-Verlag.

[24] J.D. Hamkins and R. Miller. Post's problem for ordinal register machines. In S.B. Cooper and A. Sorbi, editors, *Computation and Logic in the Real World: Proc. of CiE2007, Siena,*, volume 4497 of *Lecture Notes in Computer Science*, pages 358–367, Berlin, 2007. Springer-Verlag.

[25] J.D. Hamkins and R. Miller. Post's problem for ordinal register machines: an explicit approach. *Annals of Pure and Applied Logic*, 160(3):302–309, September 2009.

[26] J.D. Hamkins, R. Miller, D. Seabold, and S. Warner. Infinite time computable model theory. In S.B. Cooper *et al.*, editor, *New Computational Paradigms*, pages 521 – 557. Springer-Verlag, Berlin, 2008.

[27] J.D. Hamkins and D. Seabold. Infinite time Turing machines with only one tape. *Mathematical Logic Quarterly*, 47(2):271–287, 2001.

[28] J.D. Hamkins and P.D. Welch. $P^f \neq NP^f$ almost everywhere. *Mathematical Logic Quarterly*, 49(5):536–540, 2003.

[29] L. Harrington. Analytic determinacy and $0^{\#}$. *Journal of Symbolic Logic*, 43(4):684–693, 1978.

[30] H.G. Herzberger. Notes on naive semantics. *Journal of Philosophical Logic*, 11:61–102, 1982.

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

[32] K. Hrbacek and S. Simpson. On Kleene degrees of analytic sets. In H.J.Keisler J.Barwise and K.Kunen, editors, *Proceedings of the Kleene Symposium*, Studies in Logic, pages 347–352. North-Holland, 1980.

[33] G.H.C. Driscoll Jr. Metarecursively enumerable sets and their metadegrees. *Journal of Symbolic Logic*, 33:389–411, 1968.

[34] A.S. Kechris. On Spector classes. In A.S. Kechris and Y.N. Moschovakis, editors, *Cabal Seminar 76-77*, volume 689 of *Lecture Notes in Mathematics Series*, pages 245–278. Springer, 1978.

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

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

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

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

[39] S.C. Kleene. Recursive functionals and quantifiers of finite types revisited. In *Generalized Recursion Theory II, Proceedings 2nd Scandinavian Logic Symposium, Oslo, 1977*, volume 94 of *Studies in Logic and Foundations of Mathematics*, pages 185–222, Amsterdam, New York, 1978. North-Holland.

[40] A. Klev. *Magister thesis*. ILLC Amsterdam, 2007.

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

[42] P. Koepke and M. Koerwien. Ordinal computations. *Mathematical Structures in Computer Science*, 16.5:867–884, October 2006.

[43] P. Koepke and R. Miller. An enhanced theory of infinite time register machines. In A. Beckmann *et al.*, editor, *Logic and the Theory of Algorithms*, volume 5028 of *Springer Lecture Notes Computer Science*, pages 306–315. Swansea, Springer, 2008.

[44] P. Koepke and B. Seyfferth. Ordinal machines and admissible recursion theory. *Annals of Pure and Applied Logic*, 160(3):310–318, 2009.

[45] P. Koepke and B. Seyfferth. Towarda a theory of infinite time Blum-Shub-Smale machines. *CiE2012 Proceedings*, 2012.

[46] P. Koepke and R. Siders. Computing the recursive truth predicate on ordinal register machines. In A. Beckmann *et al.*, editor, *Logical Approaches to Computational Barriers*, Computer Science Report Series, page 21. Swansea, 2006.

[47] P. Koepke and P.D. Welch. A generalised dynamical system, infinite time register machines, and Π^1_1-CA_0. In B. L editor, *Proceedings of CiE 2011, Sofia*, 2011.

[48] S. Kripke. Transfinite recursion on admissible ordinals I,II. *Journal of Symbolic Logic*, 29:161–162, 1964.

[49] A Levy. Transfinite computability (abstract). *Notices of the American Mathematical Society*, 10:286, 1963.

[50] B. Löwe. Revision sequences and computers with an infinite amount of time. *Journal of Logic and Computation*, 11:25–40, 2001.

[51] B. Löwe. Space bounds for infinitary computations. In A. Beckmann *et al.*, editor, *Logical Approaches to Computational Barriers, CiE 2006*, volume 3988 of *Lecture Notes in Computer Science*, pages 319–329. Springer-Verlag, 2006.

[52] R. Lubarsky. Well founded iterations of infinite time turing machines. In R-D Schindler, editor, *Ways of Proof Theory*. Ontos, 2010.

[53] M. Machover. The theory of transfinite recursion. *Bulletin of the American Mathematical Society*, 67:575–578, 1961.

[54] M. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, 1967.

[55] A. Montalbán and R. Shore. The limits of determinacy in second order number theory. *Proceedings of the London Mathematical Society*, to appear.

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

[57] D. Normann. Turing's models. In *this volume*. 2012.

[58] R. Platek. *Foundations of Recursion Theory.* PhD thesis, Stanford, 1966.

[59] H. Putnam. Trial and error predicates and the solution to a problem of Mostowski. *Journal of Symbolic Logic*, 30:49–57, 1965.

[60] M. Rathjen. An ordinal analysis of parameter-free Π^1_2 comprehension. *Archive for Mathematical Logic*, 44(3):263–362, 2005.

[61] H. Rogers. *Recursive Function Theory.* Higher Mathematics. McGraw, 1967.

[62] G. E. Sacks and S. G. Simpson. The α-finite injury method. *Annals of Mathematical Logic*, 4:343–367, 1972.

[63] G.E. Sacks. Post's problem, admissible ordinals and regularity. *Transactions of the American Mathematical Society*, 124:1–23, 1966.

[64] G.E. Sacks. *Higher Recursion Theory.* Perspectives in Mathematical Logic. Springer Verlag, 1990.

[65] R-D. Schindler. $P \neq NP$ for infinite time Turing machines. *Monatsheft für Mathematik*, 139(4):335–340, 2003.

[66] R.-D. Schindler and M. Zeman. Fine structure theory. In M. Magidor M. Foreman, A. Kanamori, editor, *Handbook of Set Theory*, volume 1, pages 605 – 656. Springer Verlag, Heidelberg, New York, 2010.

[67] J. Shepherdson and H. Sturgis. Computability of recursive functionals. *Journal of the Association of Computing Machinery*, 10:217–255, 1963.

[68] R. A. Shore. Splitting an α recursively enumerable set. *Transactions of the American Mathematical Society*, 204:65–78, 1975.

[69] R. A. Shore. The recursively enumerable α-degrees are dense. *Annals of Mathematical Logic*, 9:123–155, 1976.

[70] S. Simpson. *Subsystems of second order arithmetic.* Perspectives in Mathematical Logic. Springer, January 1999.

[71] R.M. Solovay. Determinacy and type-2 recursion. *Journal of Symbolic Logic*, 36:374, 1971.

[72] C. Spector. Hyperarithmetic quantifiers. *Fundamenta Mathematicae*, 48:313–320, 1959.

[73] J. R. Steel. Analytic sets and Borel isomorphisms. *Fundamenta Mathematicae*, 108:83–88, 1980.

[74] G. Takeuti. On the recursive functions of ordinal numbers. *Journal of the Mathematical Society of Japan*, 12:119–128, 1960.

[75] T. Tugué. On the partial recursive functions of ordinal numbers. *Journal of the Mathematical Society of Japan*, 16:1–31, 1964.

[76] P.D. Welch. Minimality arguments in the infinite time Turing degrees. In S.B.Cooper and J.K.Truss, editors, *Sets and Proofs: Proc. Logic Colloquium 1997, Leeds*, volume 258 of *London Mathematical Society Lecture Notes in Mathematics*. C.U.P., 1999.

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

[78] P.D. Welch. The length of infinite time Turing machine computations. *Bulletin of the London Mathematical Society*, 32:129–136, 2000.

[79] P.D. Welch. Post's and other problems in higher type supertasks. In B. Löwe, B. Piwinger, and T. Räsch, editors, *Classical and New Paradigms of Computation and their Complexity hierarchies, Papers of the Conference Foundations of the Formal Sciences III*, volume 23 of *Trends in logic*, pages 223–237. Kluwer, Oct 2004.

[80] P.D. Welch. Bounding lemmata for non-deterministic halting times of transfinite Turing machines. *Theoretical Computer Science*, 394:223–228, 2008.

[81] P.D. Welch. Turing Unbound: The extent of computations in Malament-Hogarth spacetimes. *British J. for the Philosophy of Science*, 15(4):659–674, December 2008.

[82] P.D. Welch. Ultimate truth *vis à vis* stable truth. *Review of Symbolic Logic*, 1(1):126–142, June 2008.

[83] 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.

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