Non-deterministic halting times for Hamkins-Lewis Turing machines.

P.D.Welch*

March 6, 2006

In this talk we consider some issues related to the Infinite Time Turing Machine (ITTM) model of Hamkins & Lewis [3]. There a standard Turing machine (with some inessential minor modifications) is allowed to run transfinitely in ordinal time. The machine's behaviour at limit stages of time λ is completely specified by requiring that (i) the machine enter a special limit state q_L; (ii) the read/write head return to the initial starting cell at the leftmost end of the tape; (iii) the cells values - which we shall assume are taken from the alphabet of $\{0, 1\}$ - are the limsup of their previous values: that is if cell i on the tape has contents $C_i(\gamma) \in \{0, 1\}$ at time γ, then for any $i < \omega$ $C_i(\lambda) = \limsup_{\gamma \longrightarrow \lambda} \langle C_i(\gamma) | \gamma < \lambda \rangle$. The original machine specified three infinite tapes: input, scratch, and output, with a read/write head positioned over one cell from each tape simultaneously. The machine's actions at successor stages is determined by its (finite) program in the ordinary way.

A number of intriguing questions immediately spring to mind. The question of the identity of the "decidable" reals (for which $x \in 2^{\mathbb{N}}$ is there a program P_e so that on input x P_e halts on input x ("$P_e(x)\downarrow$") ?), and of the semi-decidable reals, is answered in Welch[5]. (Hamkins and Lewis [3] had previously showed, *inter alia*, that Π^1_1 predicates of reals are decidable, and that the decidable, (and semi-decidable) pointclasses of reals are strictly between Π^1_1 and Δ^1_2 in the projective hierarchy.)

We shall be concerned here rather with the question of *halting times*, or how long such a computation takes, if it is going to halt.

Definition 1 *$P_e(x)\downarrow^{\alpha}$ will denote that program $P_e(x)\downarrow$ in exactly α steps. $P_e(x)\downarrow^{\leq\alpha}, P_e(x)\downarrow^{<\alpha}$ are defined analogously.*

To clarify the above: $P_e(x)\downarrow^{\alpha}$ means that at ordinal time α the read/write head is in particular state q_s and is reading a triple of cells (one from each of the three tapes) so that it's program determines that it go into a halting state q_h. Thus a machine may halt exactly at some limit stage of time α where then $q_s = q_L$.

**Email*: p.welch@bristol.ac.uk

Suppose x is simple: perhaps it is an integer (*i.e.* it is a binary code for $n \in \mathbb{N}$ followed by an infinite string of 0's), perhaps it is 0 (in the above sense) itself. What possible halting times as e varies are there for $P_e(x)$? [3] calls an ordinal *clockable* if it is the halting time of a computation with input 0.

Further, let us define:

Definition 2 *"$P_e(x)\downarrow y$" will denote that $P_e(x)\downarrow$ and that $y \in 2^{\mathbb{N}}$ is the contents of the output tape on halting. (Again $P_e(x)\downarrow^{\alpha} y$ etc. are defined analogously).*

Then we say that y is *writable* if it is the output of some program: $P_e(0)\downarrow y$. An *ordinal* β *is writable* if some $y \in \mathrm{WO}$ is writable, and y codes a wellordering of rank β. What possible ordinals are writable? It is easy to readjust a program that demonstrates that β is writable to one that shows $\beta' < \beta$ is writable for some β'. Thus the writable ordinals are an initial segment, λ, of all ordinals. Hamkins and Lewis [3] showed that there are gaps in the clockable ordinals and the following:

Theorem 1 *Hamkins and Lewis [3] If β is admissible then it is not clockable.*

(For notions of *admissible ordinal* and *admissible set* see [1].) Welch [6] shows that λ, the suprema of the writable ordinals, is also the supremum of the clockable ordinals.

One may generalise these questions to those involving arbitrary input x. The following is Definition 24 of Deolalikar, Hamkins & Schindler [2]:

Definition 3 *An ordinal α is* nondeterministically clockable *if there is an algorithm P_e which halts in time at most α for all input and in time exactly α for some input. More generally, α is* nondeterministically clockable before β *if there is an algorithm that halts before β on all input and in time exactly α for some input.*

Symbolically: α is nondeterministically clockable iff

$$\exists e \in \mathbb{N}[\forall x \in 2^{\mathbb{N}} P_e(x)\downarrow^{\leq\alpha} \wedge \exists x \in 2^{\mathbb{N}} P_e(x)\downarrow^{\alpha}].$$

This notion arises in the paper [2], which was concerned with various complexity pointclasses defined using halting times of computations on these machines, with or without existential 'non-determinacy' witnesses.

We show the following

Theorem 2 *If β is admissible then it is not nondeterministically clockable.*

This is in fact a corollary of a more general *Bounding Lemma* (where we identify $\mathbb{R}$ with $2^{\mathbb{N}}$):

Proposition 1 *(Bounding Lemma) Suppose β be admissible. Let $F : \mathbb{R} \longrightarrow \mathbb{R}$ be an ITTM-computable total function, so that $\forall x P_e(x)\downarrow^{\leq\beta}$ where P_e computes F. Then $\exists\gamma < \beta\ \forall x P_e(x)\downarrow^{<\gamma}$.*

Let $x \in 2^{\mathbb{N}}$. Then, as is usual, we let $\omega_{1\,\mathrm{ck}}^{x}$ denote the supremum of all ordinals that are recursive in x (that is, those ordinals α with a corresponding $y \in \mathrm{WO}$ with rank of y equalling α, and the characteristic function of y is Turing recursive (in the ordinary sense of recursive) in x.

They pose the following question in [2]:

Question 6 *Suppose an algorithm halts on each input x in fewer than $\omega_{1\,\mathrm{ck}}^{x}$ steps. Then does it halt uniformly before $\omega_{1\,\mathrm{ck}}$?*

As they say an affirmative answer explains some of the phenomena observed in their paper. Perhaps somewhat remarkably this is the case (we drop the subscript ck and write ω_1^x for the first ordinal not recursive in x etc.). We prove that we have *Uniform Bounding*:

Proposition 2 *Let $F : \mathbb{R} \longrightarrow \mathbb{R}$ be ITTM-computable and total as witnessed by the program P_e. If $\forall x P_e(x)\downarrow^{<\omega_1^x}$ then $\exists \gamma < \omega_{1\,\mathrm{ck}}\ \forall x P_e(x)\downarrow^{<\gamma}$.*

We consider some further queries arising from the paper [2]. These concerned various complexity pointclasses defined using halting times of computations on Infinite Time Turing machines, with or without existential 'non-determinacy' witnesses. These classes were first explicitly introduced by Schindler in [4].

Definition 4 *Let $f : \mathbb{R} \longrightarrow \mathrm{On}$. (i) $A \in P^f$ if there is an infinite time Turing machine deciding each $x \in A$ in fewer than f(x) many steps.*

(ii) $A \in \mathrm{NP}^f$ when there is an infinite time Turing machine T such that $x \in A$ if and only if there is $y \in \mathbb{R}$ such that T accepts (x, y), and T halts on any input (x, y) in fewer than f(x) many steps.

We thus think of f as a bounding function on the number of steps needed to determine whether x is, or is not, in some pointclass A, by using some total (so always either accepting or rejecting) ITTM program. f may be a constant function, and in the case that it is with value ω^ω [2] call the pointclasses P and NP. They analyse these classes for a variety of f and show, for example:

Theorem 3 *[2] $P \neq NP \cap \text{co-}NP$.*

Concomitant with the classes P^f are the following pointclasses definable in a simple way over the $f(x)$ level of the constructible hierarchy over x :

Definition 5 $\Gamma^f = \{A \subseteq \mathbb{R} : \exists \Sigma_1 \varphi \forall x [x \in A \longleftrightarrow L_{f(x)}[x] \models \varphi[x]]\}$.

So let f be suitable such that for any $x \in \mathbb{R}$ $L_{f(x)}[x]$ is an admissible set that is a union of such. Then:

Proposition 3 *$NP^f = \Gamma^f; P^f = \Gamma^f \cap \text{co-}\Gamma^f = NP^f \cap \text{co}\, NP^f$. Thus in general NP^f does not equal the dual class $\Gamma^f \cap \text{co-}\Gamma^f$.*

This answers another of the queries of [2].

References

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

[2] V. Deolalikar, J.D. Hamkins, and R-D. Schindler. $P \neq NP \cap coNP$ for Infinite Time Turing machines. *Journal of Logic and Computation*, 15:577–592, Oct 2005.

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

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

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

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