Feedback ITTMs and Σ^0_3 Determinacy

Robert S. Lubarsky
Florida Atlantic University

Logic Colloquium 2010
Paris, France
July 25 - August 1, 2010

Introduction

How far up in the L-hierarchy do you have to go to model Σ^0_3-Determinacy?

Introduction

How far up in the L-hierarchy do you have to go to model Σ_3^0-Determinacy?
(Welch) The least model L_γ of Σ_3^0-Determinacy is between the least Σ_2-Admissible and the least Σ_2-Non-Projectible ordinals.

Introduction

How far up in the L-hierarchy do you have to go to model Σ^0_3-Determinacy?
(Welch) The least model L_γ of Σ^0_3-Determinacy is between the least Σ_2-Admissible and the least Σ_2-Non-Projectible ordinals.
Actually, Welch showed, from above, if

- $\gamma_0 < \gamma_1 < \gamma_2$
- $L_{\gamma_0} \prec_{\Sigma_2} L_{\gamma_1}$
- $L_{\gamma_0} \prec_{\Sigma_1} L_{\gamma_2}$ and
- L_{γ_2} is a limit of admissibles,

then $\gamma < \gamma_0$.

Introduction

What Welch showed from below:

Definition

β is 0-extendible if for some δ $L_\beta \prec_{\Sigma_2} L_\delta$.

Introduction

What Welch showed from below:

Definition

β is 0-extendible if for some δ $L_\beta \prec_{\Sigma_2} L_\delta$.
β is $(\alpha{+}1)$-extendible if its a Σ_2-extendible limit of α-extendibles.

Introduction

What Welch showed from below:

Definition

β is 0-extendible if for some δ $L_\beta \prec_{\Sigma_2} L_\delta$.
β is $(\alpha+1)$-extendible if its a Σ_2-extendible limit of α-extendibles.
β is κ-extendible if its a Σ_2-extendible limit of α-extendibles for each $\alpha < \kappa$.

Introduction

What Welch showed from below:

Definition

β is 0-extendible if for some δ $L_\beta \prec_{\Sigma_2} L_\delta$.
β is $(\alpha+1)$-extendible if its a Σ_2-extendible limit of α-extendibles.
β is κ-extendible if its a Σ_2-extendible limit of α-extendibles for each $\alpha < \kappa$.
β is hyperextendible if β is α-extendible for all $\alpha < \beta$.

Introduction

What Welch showed from below:

Definition

β is 0-extendible if for some δ $L_\beta \prec_{\Sigma_2} L_\delta$.
β is $(\alpha+1)$-extendible if its a Σ_2-extendible limit of α-extendibles.
β is κ-extendible if its a Σ_2-extendible limit of α-extendibles for each $\alpha < \kappa$.
β is hyperextendible if β is α-extendible for all $\alpha < \beta$.

γ is greater than the least hyperextendible.

ITTMs

(Hamkins & Lewis) An *Infinite time Turing machine* is a regular Turing machine with limit stages.

ITTMs

(Hamkins & Lewis) An *Infinite time Turing machine* is a regular Turing machine with limit stages. At a limit stage:

ITTMs

(Hamkins & Lewis) An *Infinite time Turing machine* is a regular Turing machine with limit stages. At a limit stage:

- the machine is in a dedicated state

ITTMs

(Hamkins & Lewis) An *Infinite time Turing machine* is a regular Turing machine with limit stages. At a limit stage:

- the machine is in a dedicated state
- the head is on the 0^{th} cell

ITTMs

(Hamkins & Lewis) An *Infinite time Turing machine* is a regular Turing machine with limit stages. At a limit stage:

- the machine is in a dedicated state
- the head is on the 0^{th} cell
- the content of a cell is limsup of the previous contents (i.e. 0 if eventually 0, 1 if eventually 1, 1 if cofinally alternating)

ITTMs

(Hamkins & Lewis) An *Infinite time Turing machine* is a regular Turing machine with limit stages. At a limit stage:

- the machine is in a dedicated state
- the head is on the 0^{th} cell
- the content of a cell is limsup of the previous contents (i.e. 0 if eventually 0, 1 if eventually 1, 1 if cofinally alternating)

(Welch) The latest stage at which an ITTM can enter into a loop is at the least 0-extendible (i.e. the least Σ_2-extendible).

ITTMs

(Hamkins & Lewis) An *Infinite time Turing machine* is a regular Turing machine with limit stages. At a limit stage:

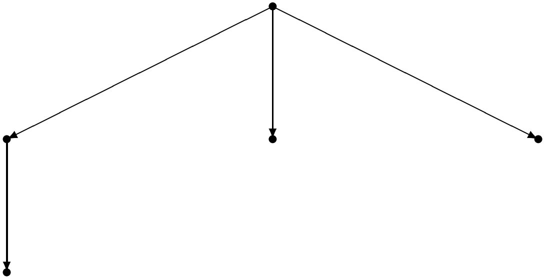
- the machine is in a dedicated state
- the head is on the 0^{th} cell
- the content of a cell is limsup of the previous contents (i.e. 0 if eventually 0, 1 if eventually 1, 1 if cofinally alternating)

(Welch) The latest stage at which an ITTM can enter into a loop is at the least 0-extendible (i.e. the least Σ_2-extendible).
(L) The least hyperextendible can be characterized with iterated ITTMs, which are machines that are allowed certain oracle calls.

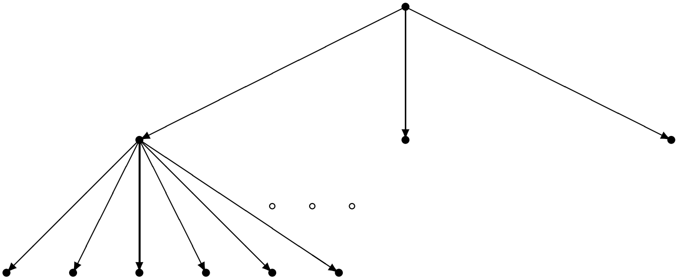
Feedback ITTMs

ITTMs with arbitrary iteration:
A computation may ask a convergence question about another computation. This can be considered calling a sub-computation. That sub-computation might do the same. This can continue, generating a *tree of sub-computations*. Eventually, perhaps, a computation is run which calls no sub-computation. This either converges or diverges. That answer is returned to its calling computation, which then continues.

Good examples



Good examples



Bad example

Bad example

One can naturally define the course of a computation if and only if the tree of sub-computations is well-founded. How is this to be dealt with?

FITTMs

Allow all possible sub-computation calls, even if the tree of sub-computations is ill-founded, and consider only those for which the tree of sub-computations just so happens to be well-founded.

FITTMs

Allow all possible sub-computation calls, even if the tree of sub-computations is ill-founded, and consider only those for which the tree of sub-computations just so happens to be well-founded. So some legal computations have an undefined result: the **freezing** computations. The **non-freezing** computations have a perfectly well-defined semantics.

FITTMs

Allow all possible sub-computation calls, even if the tree of sub-computations is ill-founded, and consider only those for which the tree of sub-computations just so happens to be well-founded. So some legal computations have an undefined result: the **freezing** computations. The **non-freezing** computations have a perfectly well-defined semantics. For the insiders: with this notion of computation, the writable, eventually writable, and accidentally writable reals are all the same.

Results

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.

Results

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Results

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

Player I is to build (the Σ_1 truth set of) a model M of "$V = L$ and $\{e\}$ converges." Player II is to find an infinite descending sequence through the ordinals in I's model.

Results

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

Player I is to build (the Σ_1 truth set of) a model M of "$V = L$ and $\{e\}$ converges." Player II is to find an infinite descending sequence through the ordinals in I's model. I has a winning strategy in V: play L_α. Hence I has a w.s. in L_γ.

Results

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

Player I is to build (the Σ_1 truth set of) a model M of "$V = L$ and $\{e\}$ converges." Player II is to find an infinite descending sequence through the ordinals in I's model. I has a winning strategy in V: play L_α. Hence I has a w.s. in L_γ. Let σ be such a w.s. Have II do nothing. If I plays L_α's truth set, we're done.

Results

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

Player I is to build (the Σ_1 truth set of) a model M of "$V = L$ and $\{e\}$ converges." Player II is to find an infinite descending sequence through the ordinals in I's model. I has a winning strategy in V: play L_α. Hence I has a w.s. in L_γ. Let σ be such a w.s. Have II do nothing. If I plays L_α's truth set, we're done. If I ever makes an assertion about M which is false for L_α, then II knows the model is non-standard, and must only find an i.d.c.

Proof continued on next slide. □

Proof

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

(continued)If I ever plays something false of L_α, then Welch showed how II can find an i.d.c. in that model, mod the following problem: in M, there could be ordinals such that

Proof

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

(continued)If I ever plays something false of L_α, then Welch showed how II can find an i.d.c. in that model, mod the following problem: in M, there could be ordinals such that

$$\beta_0 < \beta_1 < \beta_2 < \ldots < \delta_2 < \delta_1 < \delta_0, \ \beta_n \text{ standard, and } L_{\beta_n} \prec_{\Sigma_2} L_{\delta_n}.$$

Proof

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

(continued)If I ever plays something false of L_α, then Welch showed how II can find an i.d.c. in that model, mod the following problem: in M, there could be ordinals such that

$$\beta_0 < \beta_1 < \beta_2 < \ldots < \delta_2 < \delta_1 < \delta_0,\ \beta_n \text{ standard, and } L_{\beta_n} \prec_{\Sigma_2} L_{\delta_n}.$$

Any tree of sub-computations can be adorned with ordinals in a natural way. In particular, the pair β_n, δ_n is assigned to a node which is a parent to the node of $\beta_{n+1}, \delta_{n+1}$.

Proof

Theorem

If an FITTM computation converges in α-many steps, then $\alpha < \gamma$.
If an FITTM computation freezes in α-many steps, then $\alpha \leq \gamma$.

Proof.

(continued)If I ever plays something false of L_α, then Welch showed how II can find an i.d.c. in that model, mod the following problem: in M, there could be ordinals such that

$$\beta_0 < \beta_1 < \beta_2 < \ldots < \delta_2 < \delta_1 < \delta_0,\ \beta_n \text{ standard, and } L_{\beta_n} \prec_{\Sigma_2} L_{\delta_n}.$$

Any tree of sub-computations can be adorned with ordinals in a natural way. In particular, the pair β_n, δ_n is assigned to a node which is a parent to the node of $\beta_{n+1}, \delta_{n+1}$. Hence the β_n's would give an i.d.c. in $\{e\}$'s sub-computation tree, which was assumed to be well-founded. So that problem can't happen, giving II an opportunity to win, forcing I to play the truth.

Goals

Since we can't get the freezing computations themselves to be in L_γ, only initial segments of them, perhaps the ordinal of one of them is γ itself.

Goals

Since we can't get the freezing computations themselves to be in L_γ, only initial segments of them, perhaps the ordinal of one of them is γ itself.

It would also be nice to have a description of γ and of the FITTM-ordinals in terms of reflection/extendibility properties.

References

- Joel Hamkins and Andy Lewis, "Infinite Time Turing Machines," **The Journal of Symbolic Logic**, v. 65 (2000), p. 567-604
- Robert Lubarsky, "ITTMs with Feedback," in **Ways of Proof Theory** (Ralf Schindler, ed.), Ontos, 2010
- Philip Welch, "The Length of Infinite Time Turing Machine Computations," **The Bulletin of the London Mathematical Society**, v. 32 (2000), p. 129-136
- Philip Welch, "Eventually Infinite Time Turing Machine Degrees: Infinite Time Decidable Reals," **The Journal of Symbolic Logic**, v. 65 (2000), p. 1193-1203
- Philip Welch, "Characteristics of Discrete Transfinite Turing Machine Models: Halting Times, Stabilization Times, and Normal Form Theorems," **Theoretical Computer Science**, v. 410 (2009), p. 426-442
- Philip Welch, "Weak Systems of Determinacy and Arithmetical Quasi-Inductive Definitions," **JSL**, to appear