Turing degrees, The Arithmetical Hierarchy and Post's Theorem

Estifanos Getachew

December 1, 2023

Mathematical Logic and Computability Graduate Seminar
University of Waterloo

Outline

Part I: Relative Computability

- Oracle Turing Machine
- Turing Functionals
- Turing Degrees

Part II: The Arithmetical Hierarchy

Part III: Post's Theorem

Part I: Relative Computability

Part I: Relative Computability

Let us suppose we are supplied with some unspecified means of solving number-theoretic problems; a kind of oracle as it were. We shall not go any further into the nature of this oracle apart from saying that it cannot be a machine. With the help of the oracle we could form a new kind of machine (call them o-machines), having as one of its fundamental processes that of solving a given number-theoretic problem.

Alan Turing, 1939

Oracle Turing machines

Definition 1 (Oracle Turing Machine)

An Oracle Turing machine (o-machine) M is a $6-$tuple $(Q, q_0, S_1, S_2, \delta, q_f)$, where

- Q is a finite set (called the states of the M)
- $q_0 \in Q$ is the initial state.
- $S_1 = \{B, 0, 1\}$ is the *oracle* tape alphabet (which is assumed to be read-only).
- $S_2 = \{B, 1\}$ is the *working* tape alphabet.
- $\delta : Q \times S_1 \times S_2 \to Q \times S_2 \times \{L, R\} \times \{L, R\}$ is a partial function.
- $q_f \in Q$ is the final state.

Computations in o-machine

- It includes 2 two-way infinite tapes (The oracle and working tapes) dived into cells and 2 reading heads.
- Oracle tape contains the characteristics function of some set A, called the $oracle$($\in \{0,1\}^*$)
- We begin with M in the starting state q_0 scanning the leftmost cells.
- Then machine computes according to δ:
- To interpret $(q, s, t, q', t', X, Y) \in \delta$:
  - If on state q the machine reads s on the oracle tape and t on the working tape
  - it transitions into state q' replacing t with t'
  - and moves the oracle and working tape heads according to X and Y.

Computations in o-machine ctd..

- When viewed as a finite set of 7 tuples we call δ *an Oracle Turing Program.*
- Similar to register programs we fix an effective coding of all oracle Turing programs, i.e P_e is the e^{th} oracle Turing program
- The input number x is represented by a string of $x+1$ consecutive 1's.
- We say that M halts on input x if it reaches the halting state q_f in a finite number of steps.
- The output y is the total number of $1's$ in the working tape.
- M never makes any further moves after reaching state q_f
- We say u is the use of the computation if u is the maximum cell scanned on the oracle tape.

Turing Functionals

If P_e with A on its oracle tape and input x halts with output y and u is the use of the computation we write $\Phi_e^A(x) = y$ (The Turing Functional) and $\varphi_e^A(x) = u$ (The use function).

We also write:

- $\Phi_{e,s}^A(x) = y$, $\varphi_{e,s}^A(x) = u$, if it happens in s steps and $e, x, y < s$
- $\Phi_{e,s}^\sigma(x) = y$, $\varphi_{e,s}^\sigma(x) = u$ if $\sigma \in 2^{<\omega}$ ($\sigma \in \{0,1\}^*$ and is finite) on the oracle tape and $u < |\sigma|$
- $\Phi_e^A(x) \downarrow$ if $\Phi_e^A(x)$ converges (i.e P_e on input x halts)
- $\Phi_e^A(x) \uparrow$ if $\Phi_e^A(x)$ diverges.
- $W_e^A = dom(\Phi_e^A) = \{x : \Phi_e^A(x) \downarrow = y\}$
- $W_{e,s}^A = dom(\Phi_{e,s}^A) = \{x : \Phi_{e,s}^A(x) \downarrow = y\}$

Note that $\Phi_e^A(x) = y \iff \exists\, s\, \Phi_{e,s}^A(x) = y$ and $x \in W_e^A \iff \exists\, s\, W_{e,s}^A$

A-Computability, Turing reducibility

A partial function f is **Turing computable in** A *(A-Turing computable)*, written as $f \leq_T A$ if, $\Phi_e^A(x) \downarrow = y$ iff $f(x) = y$.

A set B is **Turing reducible** to A iff it's characteristics function $\chi_B \leq_T A$.

Turing reducibility generalizes the idea's in *many-one* and *one-one* reduction in standard turing machines.

In class we have effectively reduced Π_{halt} to $\{\varphi \in L_0^S : \models \varphi\}$.

Oracle Graphs of Turing Functional

Definition 2 (Minimum halting strings)

For all e, x we define the set of Minimum halting strings

$$H_{e,x} = \{\sigma : \exists\, s\, \Phi_e^\sigma(x)\downarrow \wedge\, \phi_{e,s}^\sigma(x) = |\sigma| - 1\}$$

strings σ for which the computation of $\Phi_e^\sigma(x)$ enters the halting state q_f after reading exactly σ

Definition 3 (Prefix-Free)

A set $S \subseteq 2^{<\omega}$ is said to be prefix free iff

$$\forall \sigma, \tau \in 2^{<\omega}(\sigma \in S \wedge \sigma \prec \tau) \implies \tau \notin S$$

Oracle Graphs of Turing Functional

Theorem 4

For every e and x, $H_{e,x}$ is prefix free.

Proof.

Assume $\sigma \in H_{e,x}$, then $\Phi^{\sigma}_{e,s}(x)\downarrow \wedge \phi^{\sigma}_{e,s}(x) = |\sigma| - 1$ for some s (this means that the oracle program P_e after s steps and reading exactly σ on the oracle tape entered the halting state q_f.)

Hence for any ρ such that $\sigma \prec \rho$ then $\varphi^{\rho}_{e}(x) < |\rho|$ and similarliy if $\rho \prec \sigma$ $\varphi^{\rho}_{e}(x) > |\rho|$ $\square$

Oracle Graphs of Turing Functional

Definition 5 (The Prefix-Free Oracle Graph)

Define the prefix-free oracle graph of Φ_e^σ as follows:

$$F_e = \{\langle \sigma, x, y \rangle : \Phi_e^\sigma(x) = y \wedge \sigma \in H_{e,x}\}$$

Theorem 6 (Unique Use Property)

For every x, y, e and oracle A, there is at most one $\sigma \prec A$ such that $\langle \sigma, x, y \rangle \in F_e$.

Proof.

Follows directly from $H_{e,x}$ being prefix free. □

Oracle Graphs of Turing Functionals

Definition 7 (The Oracle Graph)

Define the oracle graph of the functional Φ_e as
$G_e = \{\langle \sigma, x, y \rangle : \Phi_e^\sigma(x) = y\}$

Theorem 8 (Oracle Graph Theorem)

1. G_e *is single valued:* $(\langle \sigma, x, y \rangle \in G_e \wedge \langle \sigma, x, z \rangle \in G_e) \implies y = z$.
2. G_e *is monotonic:* $\langle \sigma, x, y \rangle \in G_e \implies \forall \tau \succ \sigma \langle \tau, x, y \rangle \in G_e$
3. $\langle \sigma, x, y \rangle \in G_e \iff \exists \tau \preceq \sigma \langle \tau, x, y \rangle \in F_e$.
4. G_e *is computably enumerable.*

Use Principle

Theorem 9 (Use Principle)

1. $\Phi_e^A(x) = y \implies \exists\, s\, \exists\sigma(\sigma \prec A \wedge \Phi_{e,s}^{\sigma}(x) = y)$
2. $\Phi_{e,s}^{\sigma}(x) = y \implies \forall t \geq s \forall \tau \succeq \sigma(\Phi_{e,t}^{\tau}(x) = y)$
3. $\Phi_e^{\sigma}(x) = y \implies \forall A \succ \sigma(\Phi_e^A(x) = y)$
4. $\Phi_{e,s}^A(x) = y \wedge A \upharpoonright\upharpoonright \varphi = B \upharpoonright\upharpoonright \varphi \implies \Phi_{e,s}^B(x) = y$, *where* $\varphi = \varphi_{e,s}^A(x)$

Proof.

(i) Any computation which converges does so at some finite stage, having used only finitely many elements

(iii) when the machine produces the output y it enters the halting state q_f, and will never make any more moves on the current input x. Therefore, for $A \succ \sigma$ the machine must eventually enter q_f and give output y by exactly the same computation, and will never make any more moves. $\square$

Computably Enumerable

Definition 10

We say B is computably enumerable in A if $B = W_e^A = dom(\Phi_e^A)$ for some e.

Definition 11

We say that B is Σ_1^A form if $B = \{x : \exists\, y_1 \ldots \exists\, y_n \; R^A(x, y_1, ..., y_n)\}$, for some A-computable relation R^A.

Theorem 12 (Quantifier Contraction)

A set B is Σ_1^A form iff there exists an A-computable relation S such that $B = \{x : \exists y S(x, y)\}$

Proof.

$(\Longleftarrow)$This holds by definition.

$(\Longrightarrow)$ Assume B is Σ_1^A form then there is an A-computable relation R^A s.t $B = \{x : \exists\, y_1 \ldots \exists\, y_n\ R^A(x, y_1, \ldots, y_n)\}$ now define S as follows:

$$S(x, z) \iff_{defn} R(x, (z)_1, (z)_2, \ldots, (z)_n)$$

where $z = p_0^{(z)_0}, p_1^{(z)_1}, \ldots, p_n^{(z)_n}$ is the prime decomposition. As the prime decomposition is a computable function ($\beta(i) = \mu n\ p_i^n \,|z \wedge p_i^{n+1} \nmid z$)we have that $S(x, z)$ is also A-computable.

Now note that

$$\begin{aligned} \exists z S(x, z) &\iff \exists z R(x, (z)_1, (z)_2, \ldots, (z)_n) \\ &\iff \exists y_1, \ldots, \exists y_n R(x, y_1, \ldots, y_n) \\ &\iff x \in B \end{aligned}$$

Hence $B = \{x : \exists z S(x, z)\}$. $\square$

Computably Enumerable in A

Theorem 13

B is computably enumerable in A iff B is Σ_1^A form.

Proof.

($\Longrightarrow$) Assume B is computably enumerable then $B = dom(\Phi_e^A)$ for some e hence:

$$\begin{aligned} x \in B &\iff x \in \{x : \Phi_e^A(x)\downarrow\} \\ &\iff \exists s \exists \sigma\, (\sigma \prec A \wedge \Phi_{e,s}^{\sigma}(x)\downarrow) \quad \text{...Use Principle} \\ &\iff \exists s \exists \sigma\, (\sigma \prec A \wedge x \in W_{e,s}^{\sigma}) \end{aligned}$$

Now note that $\sigma \prec A \iff \forall y(y < |\sigma| \implies \sigma(y) = A(y))$ but this A-computable. Also note that $x \in W_{e,s}^{\sigma}$ is a computable relation (hence A-computable). Therefore we have the form $\exists\, s\, \exists\, \sigma R(x, e, \sigma, s)$, where R is A-computable. □

Proof.

$(\Longleftarrow)$ Now assume that B is in Σ_1^A form. Then we have

$$B = \{x : \exists y_1, ..., \exists y_n R(x, y)\}$$

and hence by quantifier contraction

$$B = \{x : \exists y S(x, y)\}$$

Define the turing functional $\Phi_e^C(x) = \mu y S(x, y)$, then we have
$x \in B \iff \exists y S(x, y) \iff \Phi_e^C(x) = y \iff x \in W_e^C$ $\square$

Turing Degrees

Definition 14

(i) $A \equiv_T B$ iff $A \leq_T B$ and $B \leq_T A$. (Note. $\leq_T$ is reflexive and transitive)
(ii) Define the Turing degree (degree of unsolvability) of A, $deg(A) = \{B : B \equiv_T A\}$
(iii) Let **D** the class of all degrees then form the partial order $(\mathbf{D}, \leq)$ as $\mathbf{a} \leq \mathbf{b}$ iff $A \leq_T B$. Write $\mathbf{a} < \mathbf{b}$ iff $A <_T B$ $(A \leq_T B$ but $B \not\leq_T A)$
(iv) **a** is computably enumerable (c.e) if for some $A \in \mathbf{a}$, A is c.e.

(vi) A degree **a** is computably enumerable (c.e) in **b** if for some $A \in \mathbf{a}$, A is c.e. in some $B \in \mathbf{b}$.

(v) A degree **a** is computably enumerable in and above (c.e.a) in **b** if **a** is c.e in **b** and $\mathbf{b} \leq \mathbf{a}$

$A \equiv_T B$ should thought of as A and B being equally difficult to compute.

Turing Jumps

(i) Define the jump of A, $A^{'} = K^A = \{x : \Phi_x^A(x) \downarrow\}$. Note that this is the halting problem relativized to A.
(ii) The n^{th} jump of A as $A^{(0)} = A, A^{(n)} = (A^{(n-1)})^{'}$

Theorem 15 (Jump Theorem)

(i) A' *is c.e.a in* A
(ii) $A' \not\leq_T A$
(iii) B *is c.e in* A *iff* $B \leq_0 A'$
(iv) If A *is c.e in* B *and* $B \leq_T C$ *then* A *is c.e in* C
(v) $B \leq_T A$ *iff* $B' \leq_0 A'$
(vi) If $B \equiv_T A$ *then* $B' \equiv_1 A'$ *(i.e also* $B' \equiv_T A'$*)*
(Vii) A *is c.e in* B *iff* A *is c.e in* $\bar{B}$

Jumps of degrees

Now we are ready to define the jumps of degrees.

Definition 16

Define the jump of a degree **a**, $\mathbf{a}' = deg(A')$ for some $A \in \mathbf{a}$.

This is well defined by Jump Theorem (vi)

Let $\mathbf{0}^{(n)} = deg(\varnothing^{(n)})$. Then by the Jump Theorem (i) and (ii) we have

$$\mathbf{0} < \mathbf{0'} < .... < \mathbf{0}^{(n)}$$

Jumps of degrees

Now note that:

$$\mathbf{0} = deg(\varnothing) = \{B : B \equiv_T \varnothing\} = \{B : B \text{ is computable}\}$$

Also note that:

$$\varnothing' = K^{\varnothing} = \{x : \Phi_x^{\varnothing}(x) \downarrow\}$$

is the halting problem.

hence we have

$$\mathbf{0'} = deg(\varnothing') = \{B : B \equiv_T K\}$$

Part II: The Arithmetical Hierarchy

The Arithmetical Hierarchy

Definition 17

(i) A set B is in $\Sigma_0(\Pi_0, \Delta_0)$ iff B is computable.
(ii) For $n \geq 1$ a set B is in Σ_n if there is a computable relation $R(x, y_1, ..., y_n)$ such that:

$$x \in B \iff \exists\, y_1 \,\forall\, y_2 \,...Q y_n R(x, y_1, ..., y_n)$$

where Q is $\exists$ if n is odd and $\forall$ if n is even.

(iii) For $n \geq 1$ a set B is in Π_n if there is a computable relation $R(x, y_1, ..., y_n)$ such that:

$$x \in B \iff \forall\, y_1 \,\exists\, y_2 \,...Q y_n R(x, y_1, ..., y_n)$$

where Q is $\exists$ if n is even and $\forall$ if n is odd.
(iv) Similarly B is Δ_n iff $B \in \Sigma_n \cap \Pi_n$

Quantifier Manipulation

Definition 18

Fix a set A. If we replace everywhere "computable" in the above definition by "A-computable" then we have the definition of B being Σ_n in A ($B \in \Sigma_n^A$)

Theorem 19

(i) $A \in \Sigma_n$ *iff* $\bar{A} \in \Pi_n$
(ii) $A \in \Sigma_n\,, (or\, \Pi_n) \implies \forall\, m > n\, A \in \Sigma_n \cap \Pi_n$
(iii) $A, B \in \Sigma_n\,, (\Pi_n) \implies A \cup B, A \cap B \in \Sigma_n\,, (\Pi_n)$
(iv) $(R \in \Sigma_n \wedge n > 0 \wedge A = \{x : \exists\, y\, R(x, y)\}) \implies A \in \Sigma_n$
(vi) $(B \leq_m A \wedge A \in \Sigma_n) \implies B \in \Sigma_n$
(vii) If $R \in \Sigma_n(\Pi_n)$ *and* A, B *defined as below*

$$\langle\, x, y\,\rangle \in A \iff \forall\, z < y\, R(x, y, z)$$

$$\langle\, x, y\,\rangle \in B \iff \exists\, z < y\, R(x, y, z)$$

Then $A, B \in \Sigma_n(\Pi_n)$.

Proof.

(iii) Let $A = \{x : \exists\, y_1 \forall\, y_2 \, ... R(x, y_1, ..., y_n)\}$ and $B = \{x : \exists\, z_1 \forall\, z_2 \, ... S(x, z_1, ..., z_n)\}$. The we have:

$$\begin{aligned} x \in A \cup B \quad & \iff \quad \exists\, y_1 \forall\, y_2 \, ... R(x, y_1, ..., y_n) \vee \exists\, z_1 \forall\, z_2 \, ... S(x, z_1, ..., z_n) \\ & \iff \quad \exists\, y_1 \exists\, z_1 \,\forall\, y_2 \,\forall\, z_2 \, ... R(x, y_1, ..., y_n) \vee S(x, z_1, ..., z_n) \\ & \iff \quad \exists\, u_1 \forall\, u_2 \, ... R(x, (u_1)_0, ..., (u_n)_0) \vee S(x, (u_2)_0, ..., (u_n)_0) \end{aligned}$$

(iv) We proceed by induction on n. If $n = 0$ then R would be computable and hence A, B are computable. For $n > 0$ assume $R \in \Sigma_n$ then we have by (iv) we have $B \in \Sigma_n$. Now note:

$$\begin{aligned} \langle\, x, y \,\rangle \in A \quad & \iff \quad \forall z < y R(x, y, z) \\ & \iff \quad \forall z < y \exists u S(x, y, z, u) \quad \text{... for some } S \in \Pi_{n-1} \\ & \iff \quad \exists \sigma \forall z < y S(x, y, z, \sigma(z)) \end{aligned}$$

where σ ranges over $\omega^{<\omega}$ (finite sequences of the naturals). Now by the inductive hypothesis $\forall z < y S \in \Pi_{n-1}$, hence $A \in \Sigma_n$. $\square$

... continued

Proof.

For the case that $R \in \Pi_n$ we have by (i) $\overline{R} \in \Sigma_n$ and hence:

$$\begin{array}{rcll} \langle x, y \rangle \in \overline{A} & \iff & \langle x, y \rangle \notin \overline{A} & \\ & \iff & \neg(\forall z < y) R(x, y, z) & \text{Hence } \overline{A} \in \Sigma_n \text{ by (i)} \\ & \iff & (\exists z < y) \neg R(x, y, z) & \end{array}$$

again $A \in \Pi_n$. □

We proceed similarliy for B

Part III: Post's Theorem

Definition 20 (n-complete)

A set A is $\Sigma_n(\Pi_n)$-complete iff $A \in \Sigma_n(\Pi_n)$ and for all $B \in \Sigma_n(\Pi_n), B \leq_0 A$.

Now we relate the jump hierarchy of degrees to the arithmetical hierarchy.

Theorem 21 (Post's Theorem)

(i) $B \in \Sigma_{n+1} \iff B$ is c.e in some Π_n set $\iff$ (by The jump theorem 15(vii)) B is c.e in some Σ_n set.
(ii) $\varnothing^{(n)}$ is Σ_n-complete for $n > 0$.
(iii) $B \in \Sigma_{n+1} \iff B$ is c.e in $\varnothing^{(n)}$.
(iv) $B \in \Delta_n \iff B \leq_T \varnothing^{(n)}$.

Proof.

(i) ($\implies$) Let $B \in \Sigma_{n+1}$. The we have
$x \in B \iff \exists\, y_1\, \forall\, y_2..R(x, y_1, .., y_n)$, now define the relation

$$S(x, y_1) \iff_{defn} \quad \forall y_2 \exists y_3 ... R(x, y_1, ..., y_n)$$

now note that $S \in \Pi_n$ and also $x \in B \iff \exists\, y_1\, S(x, y_1)$. Hence B is Σ_1 in S. Therefore by Theorem 13 we have that B is c.e in S. $\square$

Proof.

$(\Longleftarrow)$ Now assume B is c.e in some Π_n set A, then by definition 10 we have for some e: $B = W_e^A = \{x : \Phi_e^A(x)\downarrow\}$ hence:

$$\begin{array}{rcll} x \in B & \iff & \Phi_e^A(x)\downarrow & \\ & \iff & \exists s \exists \sigma(\sigma \prec A \wedge \Phi_{e,s}^{\sigma}(x)\downarrow) & \text{...by Use Principle} \\ & \iff & \exists s \exists \sigma(\sigma \prec A \wedge x \in W_{e,s}^{\sigma}) & \end{array}$$

As the relation $x \in W_{e,s}^{\sigma}$ is computable, by Theorem 19(iv) it suffices to show that $\sigma \prec A$ is Σ_{n+1}. Now note that:

$$\begin{array}{rcl} \sigma \prec A & \iff & \forall y < |\sigma|(\sigma(y) = A(y)) \\ & \iff & \forall y < |\sigma|((\sigma(y) = 1 \wedge y \in A) \vee (\sigma(y) = 0 \wedge y \notin A)) \end{array}$$

Now note the relation $\sigma(y) = 1$ is computable , $y \in A$ is in Σ_n, $y \notin A$ in Π_n, hence by Theorem 19 (ii) they are all in Σ_{n+1}. Hence again by 19 (iii) we have the form $\forall y < |\sigma| R(y, \sigma)$ for $R \in \Sigma_{n+1}$. This is bounded quantification hence by Theorem 19 (vi), we have $\sigma \prec A$ is Σ_{n+1}. □

Proof.

(ii) We proceed by induction on n. Now for $n = 1$ we have $\varnothing^{(1)} = \{x : \Phi_x^{\varnothing}(x)\downarrow\}$, and this is just the halting problem, which is computably enumerable and 1-complete (Theorem 2.4.2. on Soare's book). Hence is Σ_1-complete.

For $n \geq 1$, Assume $\varnothing^{(n)}$ is Σ_n-complete then we have by The jump theorem 15(i) we have that $\varnothing^{(n+1)}$ is c.e in $\varnothing^{(n)}$, hence by (i) we have $\varnothing^{(n+1)} \in \Sigma_{n+1}$.

Now note that:

$$\begin{aligned} B \in \Sigma_{n+1} \quad & \iff \quad B \text{ c.e in some } \Sigma_n \text{set by (i)} \\ & \iff \quad B \text{ is c.e in } \varnothing^{(n)} \quad \text{...induction and jump theorem 15(iv)} \\ & \iff \quad B \leq_1 \varnothing^{(n+1)} \quad \text{...jump theorem 15(iii)} \end{aligned}$$

Hence $\varnothing^{(n+1)}$ is Σ_{n+1}-complete $\square$

Proof.

(iii)

$$\begin{aligned} B \in \Sigma_{n+1} &\iff B \text{ c.e in some } \Sigma_n \text{set by (i)} \\ &\iff B \text{ is c.e in } \varnothing^{(n)} \quad \text{...(ii) and jump theorem 15(iv)} \end{aligned}$$

(iv)

$$\begin{aligned} B \in \Delta_{n+1} &\iff B \in \Sigma_{n+1} \cap \Pi_{n+1} \\ &\iff B, \overline{B} \in \Sigma_{n+1} \\ &\iff B, \overline{B} \text{ c.e in } \varnothing^{(n)} \quad \text{...by (iii)} \\ &\iff B \leq_T \varnothing^{(n)} \end{aligned}$$

$\square$

An obvious consequence of the above is that $\varnothing^{(n+1)} \in \Sigma_{n+1} - \Sigma_n$. Hence the arithmetical hierarchy does not collapse.

Corollary 22

$\varnothing^{(n+1)} \in \Sigma_{n+1} - \Sigma_n$.

Proof.

Assume for the sake of contradiction that $\varnothing^{(n+1)} \in \Sigma_n$ then as $\varnothing^{(n)}$ is Σ_n-complete we have $\varnothing^{(n+1)} \leq_1 \varnothing^{(n)}$, contradicting the jump theorem (Theorem 15 (ii)). $\square$