Notes on Computability Theory

Edward Kim

Contents

1 Undecidibility

1.1 Halting Problem

Given any register program P consisting of a finite alphabet($\mathbb{A}$), we can assign a code to represent the program through a Gödel numbering scheme. Let ξ_P be the Gödel number of P. Since we have a finite alphabet and every program can be represented as a finite string of symbols, the number of possible programs must be countable. Thus, the set $\Pi = \{\xi_P | P \text{ is a valid program over } \mathbb{A}\}$ is a countable set. Checking the syntax of programs is computable through a set of rules governing said syntax. Thus, it must be said that Π is a computable set. We now arrive at on of the cornerstone theorems in Computability Theory:

Theorem 1.1. *(Halting Problem)*

The set $\Pi'_{halt} = \{\xi_P | P \textit{ is a valid program over } \mathbb{A} \textit{ and } \xi_P \to halt\}$ *is not computable.*

Proof. Suppose that there existed a program P_0 which decided this set. Then for all P

$$P_0 : \xi_P \to \square \text{ if } P : \xi_P \to halt$$
$$P_0 : \xi_P \to \eta \text{ for some } \eta \neq \square \text{ if } P : \xi_P \to \infty$$

Create any program P_1 that does the direct opposite of P_0:

$$P_0 : \xi_P \to \infty \text{ if } P : \xi_P \to halt$$
$$P_0 : \xi_P \to halt \text{ if } P : \xi_P \to \infty$$

Now suppose we plugged in P_1 into itself, then $P_1 : \xi_P \to \infty$ iff $P_1 : \xi_P \to halt$. This is a contradiction.

□

Lemma 1.1. *The set* $\Pi_{halt} = \{\xi_P | P \textit{ is a valid program over } \mathbb{A} \textit{ and } \square \to halt\}$ *is not computable.*

Lemma 1.2. Π_{halt} *is recursively enumerable.*

Proof. Enumerate through n=1,2,3.... and generate all programs whose Gödel numbers are $\leq n$. Run each one for at most n steps with input $\square$. If a program in the list halts within that time, enumerate the program. □

We can think of the halting set through a different lens. Let φ_e denote the e^{th} program. We can also enumerate the domains of $\varphi_0, \varphi_1, ...$ through a similar tactic as the above lemma. Let $W_0, W_1, ...$ denote the domains of these functions. Since not all of these functions are total, we see that $W_e \subseteq \mathbb{N}$. The halting set can be described as $K = \{x | x \in W_x\}$. This is akin to taking all programs which halt on their own Gödel number.

1.2 Undecidibility of First-Order Logic

Theorem 1.2. *(Church's Theorem) Let* φ *be a first-order sentence. Then the set* $\{\varphi | \models \varphi\}$ *of valid first-order sentences is not computable.*

1.3 Undecidibility of Arithmetic

2 Peano Arithmetic and its Models

2.1 Encoding Proofs into PA

3 Gödel's Incompleteness Theorems

Theorem 3.1. *(Fixed Point Theorem) Suppose Φ is a set of sentences such that $PA \subseteq \Phi$. For every formula $\psi(x)$ with one free variable, there is a sentence φ such that*

$$\Phi \vdash \varphi \leftrightarrow \psi(n^{\varphi})$$

Intuitively, the theorem states that there exists a formula in arithmetic φ such that φ proves that it has property ψ.

Proof. Let us define a computable function $F : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ by the following:

$$F(n,m) = \begin{cases} n^{\psi(\bar{m})} \text{ if } n \text{ is the code for } \psi \\ 0 \text{ otherwise} \end{cases}$$

We see that this function gives us the code for the formula where $\bar{m}$ replaces the free variable in ψ. Since this is a computable function, it is representable in PA. We wee that PA allows us to encode finite sequences with natural numbers by the β-function encoding scheme.

Now suppose that F is represented by the formula $\alpha(x,y,z)$ where $\alpha(x,y,z)$ says that $F(x,y) = z$. Define

$$\beta(x) := \forall z(\alpha(x,x,z) \to \psi(z))$$
$$\varphi := \forall z(\alpha(\bar{n^{\beta}}, \bar{n^{\beta}}, z) \to \psi(z))$$

φ states that ψ holds at $\beta(\bar{n^{\beta}})$. Now let us check if $\varphi \to \psi(\bar{n^{\beta}})$.

Since $PA \vdash \alpha(\bar{n^{\beta}}, \bar{n^{\beta}}, \bar{n^{\varphi}})$, $\Phi \vdash \alpha(\bar{n^{\beta}}, \bar{n^{\beta}}, z) \to z = \bar{n^{\varphi}}$ since $n^{\beta(\bar{n^{\beta}})} = \bar{n^{\varphi}}$.

Thus, $\Phi \vdash \forall z(\alpha(\bar{n^{\beta}}, \bar{n^{\beta}}, z) \to \psi(z)) \to \psi(\bar{n^{\varphi}})$ which is exactly $\Phi \vdash \varphi \to \psi(\bar{n^{\beta}})$.

Conversely, since F is representable in Φ, $\Phi \vdash \exists^{=1}\alpha(\bar{n^{\beta}}, \bar{n^{\beta}}, z)$. The previous tells us that $\Phi \vdash \alpha(\bar{n^{\beta}}, \bar{n^{\beta}}, z) \to z = \bar{n^{\varphi}}$.

Since $\psi(\bar{n^{\varphi}})$ is the hypothesis, we get that $\Phi \vdash \psi(\bar{n^{\varphi}}) \to \forall z(\alpha(\bar{n^{\beta}}, \bar{n^{\beta}}, z) \to \psi(z))$ which is exactly $\Phi \vdash \psi(\bar{n^{\beta}}) \to \varphi$. □

We can use the Fixed Point Theorem to show Tarski's Theorem.

Theorem 3.2. *(Tarski's Theorem) $Th(\mathbb{N})$ is not definable within $Th(\mathbb{N})$.*

Proof. Suppose there existed a formula in arithmetic θ such that $\mathbb{N} \vDash \theta(\bar{n^{\varphi}})$ iff $\mathbb{N} \vDash \varphi$. Consider the sentence for $\neg\theta(x)$. By Fixed Point Theorem, there exists a formula φ such that

$$PA \vdash \varphi \leftrightarrow \neg\theta(\bar{n^{\varphi}})$$

But then, by definition,

$$\mathbb{N} \vDash \varphi \leftrightarrow \neg\theta(\bar{n^{\varphi}})$$

Thus, $\mathbb{N} \vDash \varphi$ iff $\mathbb{N} \vDash \theta(\bar{n^{\varphi}})$ iff $\mathbb{N} \vDash \neg\varphi$ which contradicts the consistency of $Th(\mathbb{N})$. Thus, $\{\bar{n^{\varphi}} : \mathbb{N} \vDash \varphi\}$ is not definable. □

3.1 Gödel's First Incompleteness Theorem

Suppose that we had a computable theory $PA \subseteq \Phi$. Since this theory extends PA, we can encode finite sequences as numbers within $\mathbb{N}$. This even includes finite proofs which are just sequences of formuala which themselves can be encoded as sequences. Let $(\theta_1, \theta_2, ..., \theta_k)$ be a sequence of proofs. Then to tell whether $(\theta_1, \theta_2, ..., \theta_k)$ is a proof from Φ is computable (check if each steps comes from the previous ones or from Φ inductively). Thus, to tell whether a number is the code of a proof from Φ is computable. We illustrate this by the following function:

$$P(n, m) \leftrightarrow \text{ } n \text{ is the code of a proof of } \varphi \text{ and } m = n^{\varphi}$$

By the observations above P is a computable function and thus representable in PA. Let $\xi(n, m)$ be the formula representing P in Φ

Observe that the set $n^{\varphi} : \Phi \vdash \varphi$ is exactly described by a Σ_1^0 statement of the form:

$$\exists x \xi(x, n^{\varphi})$$

Now let us use Fixed Point Theorem to create a formula which states that the property: I am not provable.

$$\Phi \vdash \varphi \leftrightarrow \neg\exists x \xi(x, n^{\varphi})$$

Theorem 3.3. *(Gödel's First Incompleteness Theorem) Suppose $PA \subseteq \Phi$ is a computable and consistent theory. Then there exists a sentence in arithmetic φ such that $\Phi \nvdash \varphi$ and $\Phi \nvdash \neg\varphi$. Thus, PA is incomplete.*

Proof. Suppose that $\Phi \vdash \varphi$. Then $PA \vdash \exists x \xi(x, \bar{n^{\varphi}})$. So $\Phi \vdash \neg\varphi$. This contradicts the consistency of Φ. A similar argument can be made for $\Phi \vdash \neg\varphi$. □

Hence, if Φ is consistent, then $\Phi \nvdash \varphi$ and so $PA \vdash \neg\exists\xi(x, \bar{n^{\varphi}})$ so the Σ_1^0 sentence holds in $\mathbb{N}$.

Gödel's Second Incompleteness Theorem is derived by noticing the following claim:

Claim: The theorem "If $PA \subseteq \Phi$ and Φ is consistent, then $\Phi \nvdash \varphi$ and $\Phi \vdash \varphi \leftrightarrow \bar{n^{\varphi}}$ is not provable in Φ" is provable in PA.

In other words, Gödel's First Incompleteness Theorem can be proved within PA.

Theorem 3.4. *(Gödel's Second Incompleteness Theorem) Suppose $PA \subseteq \Phi$, Φ is consistent and computable then $\Phi \nvdash CON(\Phi)$.*

Proof. Suppose that we encode $CON(\Phi)$ be a formula that states that there exists no code for a program which provides a proof that $0 = 1$. From the previous result, we see that

$$\Phi \vdash \text{"If } \Phi \text{ is consistent, then } \Phi \nvdash \varphi\text{"}$$

but this is precisely,

$$\Phi \vdash \text{"If } \Phi \text{ is consistent, then } \varphi\text{"}$$

from our Gödel sentence. Hence, if Φ is consistent, then (since $\Phi \nvdash \varphi$),

$$\Phi \nvdash \text{"}\Phi \text{ is consistent"}$$

□

In particular, $PA \nvdash CON(PA)$. However, then $PA \cup \neg CON(PA)$ is a consistent theory. Since $PA \cup \neg CON(PA)$ is consistent, by Gödel's Completeness Theorem, there exists a model $\mathcal{M}$ such that $\mathcal{M} \vDash PA \cup \neg CON(PA)$. It follows that $\mathcal{M}$ is also a non-standard model of PA since $\mathcal{M} \vDash PA$. In addition, there must be a proof from PA that leads to the conclusion of PA's inconsistency. If PA were consistent, then every number $n \in \mathbb{N}$ would encode a valid well-founded finite proof in PA. Since PA is consistent, no proofs in the initial segment of $\mathcal{M}$ would code the proof of a statement ψ and its negation $\neg\psi$. Thus, it must be said that the proof must exist in the non-standard portion of $\mathcal{M}$. Thus, a non-standard elements a must encode the proof of $\neg CON(PA)$.

3.2 Rosser's Trick and Self-Referential Statements

Let $PA \subseteq \Phi$ be a computable and consistent theory. Define the Rosser Sentence to be represented by the following computable function:

$$R(x,y) = \text{“}x \text{ codes for a proof of } y \text{ and for any code } z < x,\ z \text{ does not encode a proof of } \neg y\text{”}$$

Let φ be a formula such that, by the Fixed Point Theorem,

$$\Phi \vdash \varphi \leftrightarrow \neg\exists x R(x, \bar{n^{\varphi}})$$

φ states that if there exists a Φ-proof of me then there exists an even smaller Φ-proof of my negation.

Theorem 3.5. *(Gödel-Rosser Theorem) Let $PA \subseteq \Phi$ be a computable and consistent theory and let φ be defined as above. Then $\Phi \nvdash \varphi$ and $\Phi \nvdash \neg\varphi$.*

Proof. Suppose that $\Phi \vdash \varphi$. Then let n be the smallest code encoding a proof of φ. Since Φ is consistent, we see that $\Phi \nvdash \neg\varphi$. However, by PA axioms,

$$PA \vdash \forall z(z < \bar{m} \to z = \bar{0} \vee z = \bar{1} \vee ... \vee z = m \bar{-} 1)$$

Thus, through PA, we can verify computably that every code less than n does not match a proof of $\neg\varphi$. Thus, there exist no smaller proofs of $\neg\varphi$. However, this is exactly $\Phi \vdash \neg\varphi$.

Now suppose that $\Phi \vdash \neg\varphi$. Then, by a similar argument, let n be the smallest code representing the proof of $\neg\varphi$. By consistency of Φ, no smaller code $z < n$ can encode a proof for φ. This can be systematically checked through PA. Thus,

$$PA \vdash z \text{ does not code a proof of } \varphi \text{ if } z < n$$

Suppose that $z \geq n$ then:

$$PA \vdash (\exists n \leq z)\ n \text{ codes a proof of } \neg\varphi$$

Thus,

$$PA \vdash (\forall z) z \text{ does not code a proof of } \varphi \vee (\exists n \leq z)\ n \text{ codes a proof of } \neg\varphi$$
$$PA \vdash (\forall z)\neg(n \text{ codes a proof of } \varphi \wedge (\exists z < y)\ n \text{ does not code a proof of } \neg\varphi)$$

However, this forces PA to prove that:

$$PA \vdash \text{If } x \text{ is a proof of } \varphi \text{ from } \Phi, \text{ then } x \text{ has a smaller proof of } \neg\varphi \text{ from } \Phi$$

Therefore, $\Phi \vdash \varphi$. These results both contradict the consistency of Φ. $\square$

Theorem 3.6. *(Rosser) If $PA \subseteq \Phi$ is a computable and consistent, there are formulas φ_1 and φ_2 such that $\Phi \cup \{\varphi_1\} \nvdash \varphi_2$ and $\Phi \cup \{\varphi_2\} \nvdash \varphi_1$.*

Proof. Let φ_1 be the formula which states that for any Φ-proof of φ_1, there is a smaller Φ-proof of $\neg\varphi_1$. Also, let φ_2 be a formula which states that for any $(\Phi \pm \varphi_1)$-proof of φ_2, there exists a smaller $(\Phi \pm \varphi_1)$-proof of $\neg\varphi_2$. We know from Rosser's Trick that $\Phi \nvdash \varphi_1$ and $\Phi \nvdash \neg\varphi_1$. For the sake of contradiction, suppose that $\Phi \pm \varphi_1 \vdash \varphi_2$. We deduce that $\Phi \vdash \pm\varphi_1 \to \varphi_2$. Thus, PA can encode the proof of $\pm\varphi_1 \to \varphi_2$ through a natural number since there exists a valid finite proof of Φ. Thus,

$$PA \vdash \text{" There is a proof of } \pm\varphi_1 \to \varphi_2\text{"}$$

PA can computably check through all smaller numbers (proofs).

<u>Case 1:</u> There exists a smaller Φ-proof of $\pm\varphi_1 \to \neg\varphi_2$.

Then there exists two possibilities:

<u>Possibility 1</u>: Same boolean version of φ_1 appears twice. Then we have $\varphi_1 \to \varphi_2$ and $\varphi_1 \to \neg\varphi_2$. Then $\Phi \vdash (\varphi_1 \to \neg\varphi_2) \wedge (\varphi_1 \to \varphi_2)$. This is a contradiction. thus, $\Phi \vdash \neg\varphi_1$. This is impossible by our previous analysis (Rosser's Trick).

<u>Possibility 2</u>: Otherwise: $\varphi_1 \to \varphi_2$ and $\neg\varphi_1 \to \neg\varphi_2$. Thus $\Phi \vdash \varphi_1 \leftrightarrow \varphi_2$. By our Case 1 assumption, $PA \vdash \varphi_2$. Thus, $PA \vdash \varphi_1$. This is again impossible by our previous analysis.

<u>Case 2:</u> There does not exist a smaller proof of $\pm\varphi_1 \to \varphi_2$.
Then by inspection of the smaller numbers less than our smaller proof, we conclude that

$$PA \vdash \text{ " There is a } \Phi\text{-proof of } \pm\varphi_1 \to \varphi_2 \text{ with no smaller proof of } \pm\varphi_1 \to \neg\varphi_2\text{"}$$

However, this is exactly $PA \vdash \neg\varphi_2$. Thus, $\Phi \pm \varphi_1 \vdash \neg\varphi_2$. From our original assumption that $\Phi \pm \varphi_1 \vdash \varphi_2$, we have that $\Phi \pm \varphi_1$ is inconsistent. Thus, $\Phi \vdash \varphi_1$ or $\Phi \vdash \neg\varphi_1$. This contradicts our previous analysis once again. □

Theorem 3.7. *(Rosser) There exist sentences φ_1 and φ_2 such that*

$$PA + CON(PA + \varphi_1) \nvdash CON(PA + \varphi_2)$$
$$PA + CON(PA + \varphi_2) \nvdash CON(PA + \varphi_1)$$

4 Some Recursion Theory

Theorem 4.1. *(S^m_n-Theorem) Given m,n, there is a primitive recursive, one-to-one function $S^m_n(e, x_1, ..., x_n)$ such that*

$$\varphi_{S^m_n(e,x_1,...,x_n)}(y_1, ..., y_m) = \varphi_e(x_1, ..., x_n, y_1, ..., y_m)$$

The S^m_n-Theorem intuitively states that data can be incorporated into a program. However, data can encode programs themselves, and thus incorporating them into a program could be seen as incorporating subroutines.

Theorem 4.2.

5 Arithmetic Hierarchy

Definition Let φ be a formula in first-order arithmetic. φ is called Δ_0 if it contains only bounded quantifiers. In other words, φ is of the form $\exists(x \leq t)\theta(x, y)$ or $\forall(x \leq t)\theta(x, y)$. These are shorthand forall $\exists x(x \leq t \wedge \theta(x, y))$ and $\forall x(x \leq t \to \theta(x, y))$ respectively.

Definition We classify a first-order arithmetic formula ψ as Σ_n^0 if it is of the form $\exists x\theta$ where θ is a Π_{n-1}^0 formula. Likewise, we classify ψ as Π_n^0 if it is of the form $\forall x\theta$ where θ is a Σ_{n-1}^0 formula.

Definition We classify a formula as Δ_n^0 if there exists a Σ_n^0-formula ψ_1 and a Π_n^0-formula ψ_2 that are both equivalent to the formula.

Because we can add redundant quantifiers, Σ_n^0 formulas can be classified as Σ_m^0 formulas and Π_n^0 formulas can be classified as Π_m^0 where $m > n$.

In other words, a Σ_1^0 formula is equivalent to a formula that begins with an existential quantifer and alternates between $\exists$ and $\forall$ $n - 1$ times. Π_1^0 is defined similarly. Also, we notice that a formula ρ is Π_n^0 if $\neg\rho$ is Σ_n^0.

5.1 Definability of sets in $\mathbb{N}$

Suppose we had a set X that is defined by some formula in first-order arithmetic φ. Then, by defintion of definability, we would have the following result:

$$a \in X \leftrightarrow \mathbb{N} \vDash \varphi(\bar{a})$$

Let φ be a Δ_0 formula, the subset of natural numbers $\{n : \varphi(n)\}$ is computable. This follows from the defintion of φ being bounded. This allows us to test a finite number of natural numbers to conclude whether the formula holds for any n. Thus, it follows that:

Claim: For any Σ_1^0 formula ψ, the set $\{n : \psi(n)\}$ is recursively enumerable. The converse holds as well.

Proof. Suppose that φ is a Σ_1^0 formula of the form:

$$\varphi := \exists x_1 ... \exists x_k \theta(x_1, ..., x_k)$$

where θ is a Δ_0 formula. To enumerate through all witnesses, enumerate through all tuples $(x_1, ..., x_k)$ and output any that satisfy θ. Conversely, suppose that a set is recursively enumerable. We will prove that this set can be described by a Σ_1^0-formula. Since the set is r.e, there exists a program which enumerates through all elements of this set. However, we know that the steps of these computations can be captured by PA through the β-function. However, the formula representing the graph of the function is bounded and we can formulate a formula that says that there exists a parameters to the function that encode such a computation of a program. This is precisely a Σ_1^0-formula. $\square$

From the above result, it follows that a Π_1^0 formula defines a co-recursively enumerable set.

5.2 Turing Reducibility and Degrees

Definition A relation or function on $\mathbb{N}$ is computable relative to $X \subseteq \mathbb{N}$ if there exists an algorithm to compute the relation/function in a register machine augmented by the operation which returns membership information about X.

Definition If set Y is computable relative to X, then Y is Turing reducible to X or $Y \leq_T X$.

This is synonymous to an **Oracle Turing Machine**.

We also see that Turing reducibility gives us a partial order since

$$X \leq_T X$$
$$(X \leq_T Y) \wedge (Y \leq_T Z) \to X \leq_T Z$$

Definition Two sets X, Y are Turing equivalent if $X \leq_T Y$ and $Y \leq_T X$. This is denoted by $X \equiv_T Y$. The equivalence classes are deemed Turing degrees (or degrees of unsolvability).

<u>Claim:</u> If X is computable from R and R is computable, then X is computable

Proof. If the program X is guarenteed to halt when consulting R and R will always give an answer, then X is guarenteed to be compuatable. ☐

Definition Let $\mathcal{K}^X = \{x | x \in W_x^X\}$ be the jump of X or X'. This set is equivalent to the halting set of the oracle machine assigned to X. Let $X^{(n)}$ denote the n^{th} jump of X where $X^{(0)} = X$ and $X^{(n+1)} = (X^{(n)})'$

Theorem 5.1. *(Post) If A is any r.e set then $A \leq_T \mathcal{K}$*

Proof. Let prove that there exists a recursive function f such that

$$x \in A \leftrightarrow f(x) \in \mathcal{K} \leftrightarrow f(x) \in \mathcal{W}_{f(x)}$$

By S_m^n theorem, let f be a recursive function such that

$$\mathcal{W}_{f(x)} = \begin{cases} \omega \text{ if } x \in A \\ \emptyset \text{ otherwise} \end{cases}$$

Then we see that:

- $x \in A \to \mathcal{W}_{f(x)} = \omega \to f(x) \in \mathcal{W}_{f(x)} \to \mathcal{W}_{f(x)} \in \mathcal{K}$
- $f(x) \in \mathcal{K} \to f(x) \in \mathcal{W}_{f(x)} \to \mathcal{W}_{f(x)} \neq \emptyset \to x \in A$

☐

Definition A set A is Σ_n^0-complete if it is Σ_n^0 and for every Σ_n^0 set B, $B \leq_T A$. Π_n^0-complete sets are defined similarly.

Theorem 5.2. *for $n > 1$ the following are equivalent*

1. *S is Σ_n^0*
2. *S is recursively enumerable relative to some Π_{n-1}^0 set.*

3. *S is recursively enumerable relative to some* Δ^0_n *set.*

Proof. $1 \to 2$: This follows from the fact that a Σ^0_n formula can be written as $\exists x_1 \theta(\bar{y}, x_1)$ where $\theta(x_1, \bar{y})$ is a Π^0_{n-1} formula. Thus, we can test for each potential witness $(\bar{y}, a)$, if $\theta(\bar{y}, a)$ is satisfiable by asking the if $\bar{y} \in R$ where R is the set defined by θ.

$2 \to 3$: This follows since any Π^0_{n-1} set is a Δ^0_n (add redundent quantifiers) and any tuple of relations each Π^0_{n-1} or Σ^0_{n-1} can be replaced by a single Δ^0_n set(?).

$3 \to 1$: Suppose that S is computably enumerable relative to A where A is a Δ^0_n set. Let $\Phi(x, y)$ be a partial function using an oracle machine attached to A. We can computably enumerate S by testing values on Φ. In other words, $S = W^A_e$. If we can enumerate $a \in \mathbb{N}$, then there exists a halting computation c (according to the oracle machine) such that at certain points $q_1, ...q_k$ when queried to A give values $v_1, .., v_k$ when given then input a. We see that the statement $q_i \in A$ can be described by a Σ^0_n formula. Also, $q_i \notin A \leftrightarrow q_i \in \neg A$ is also a Σ^0_n statement. □

Theorem 5.3. *(Post's Theorem)*

i. $B \in \Sigma^0_{n+1}$ *if and only if it is recursively enumerable in some* Σ^0_n *or some* Π^0_n*;*

ii. $\emptyset^{(n)}$ *is a complete* Σ^0_n *set;*

iii. $B \in \Sigma^0_{n+1} \leftrightarrow B$ *is recursively enumerable in* $\emptyset^{(n)}$*;*

iv. $B \in \Delta^0_{n+1} \leftrightarrow B \leq_T \emptyset^{(n)}$.

Proof. i. The proof is similar to one given in Theorem 5.2

ii. We proceed by induction. We know that by 5.1 that the case for $n = 1$ is true. Assume that the assumption holds for some n. Now suppose we have a Σ^0_{n+1} set P. We know that we can transform this set can be expressed by $P(\vec{x}) = \exists y R(\vec{x}, y)$ where $R \in \Pi^0_n$. Thus, R is recursively enumerable relative to a Π^0_n statement. We can easy use $\neg R$ as an oracle to achieve the same results (simply ask if the elements you seek is not in $\neg R$). By that token, R is recursively enumerable to a Σ^0_n statement B. However, by our induction hypothesis, we know that $B \leq_T \emptyset^{(n)}$. Thus, P is recursively enumerable to $\emptyset^{(n)}$. By a relativized verison of 5.1, we see that $P \leq_T \emptyset^{(n+1)}$. Now $\emptyset^{(n+1)}$ is also a Σ^0_{n+1} set as well since $\emptyset^{(n+1)}$ is recursively enumerable to $\emptyset^{(n)}$ (by Turing jump). However, by induction, $\emptyset^{(n)}$ is a Σ^0_n set. By part (i), it must be said that $\emptyset^{(n+1)}$ is a Σ^0_{n+1} set.

iii. If $B \in \Sigma^0_{n+1}$ then B is recursively enumerable in some Π^0_n set. However, if we take the negation of this set, we can compute B by using a Σ^0_n set which , by part (ii), is computable by $\emptyset^{(n)}$.

iv. If $B \in \Delta^0_{n+1}$ then $B, \bar{B} \in \Sigma^0_{n+1}$. Thus, $B, \bar{B}$ is recursively enumerable $\emptyset^{(n)}$ and so $B \leq_T \emptyset^{(n)}$. □

Theorem 5.4. *If A is recursively approximatable iff* Δ^0_2 *iff* $R \leq_T \mathcal{K}$