Turing Degrees and the Friedberg-Muchnik Theorem

Mathematical Logic for Computer Science

Master's Degree in Computer Science

Simone Bianco (1986936)

Academic Year 2024/2025

SAPIENZA
UNIVERSITÀ DI ROMA

Table of Contents

Introduction

Notation

Let $\mathcal{M} = \{M_0, M_1, M_2, \ldots\}$ be the set of all Turing Machines working on $\Sigma = \{0, 1\}$. We denote with $\Phi^A_{i,s}(x)$ the **computation** of M_i for $s \in \mathbb{N}$ steps on input $x \in \mathbb{N}$ while having access to an oracle for the subset $A \subseteq \mathbb{N}$

- If the computation halts after s steps, we write $\Phi^A_{i,s}(x) \downarrow$.
- When the computation halts for some s, we simply write $\Phi^A_i(x) \downarrow$.
- If no value s exists, we write $\Phi^A_i(x) \uparrow$.
- We denote with $\phi^A_i(x)$ the **output** of the computation (defined iff $\Phi^A_i(x) \downarrow$)

Introduction

Notation

Let $\mathcal{M} = \{M_0, M_1, M_2, \dots\}$ be the set of all Turing Machines working on $\Sigma = \{0, 1\}$. We denote with $\Phi^A_{i,s}(x)$ the **computation** of M_i for $s \in \mathbb{N}$ steps on input $x \in \mathbb{N}$ while having access to an oracle for the subset $A \subseteq \mathbb{N}$

- If the computation halts after s steps, we write $\Phi^A_{i,s}(x) \downarrow$.
- When the computation halts for some s, we simply write $\Phi^A_i(x) \downarrow$.
- If no value s exists, we write $\Phi^A_i(x) \uparrow$.
- We denote with $\phi^A_i(x)$ the **output** of the computation (defined iff $\Phi^A_i(x) \downarrow$)

Introduction

Notation

Let $\mathcal{M} = \{M_0, M_1, M_2, \ldots\}$ be the set of all Turing Machines working on $\Sigma = \{0, 1\}$. We denote with $\Phi^A_{i,s}(x)$ the **computation** of M_i for $s \in \mathbb{N}$ steps on input $x \in \mathbb{N}$ while having access to an oracle for the subset $A \subseteq \mathbb{N}$

- If the computation halts after s steps, we write $\Phi^A_{i,s}(x) \downarrow$.
- When the computation halts for some s, we simply write $\Phi^A_i(x) \downarrow$.
- If no value s exists, we write $\Phi^A_i(x) \uparrow$.
- We denote with $\phi^A_i(x)$ the **output** of the computation (defined iff $\Phi^A_i(x) \downarrow$)

Introduction

Notation

Let $\mathcal{M} = \{M_0, M_1, M_2, \ldots\}$ be the set of all Turing Machines working on $\Sigma = \{0, 1\}$. We denote with $\Phi_{i,s}^A(x)$ the **computation** of M_i for $s \in \mathbb{N}$ steps on input $x \in \mathbb{N}$ while having access to an oracle for the subset $A \subseteq \mathbb{N}$

- If the computation halts after s steps, we write $\Phi_{i,s}^A(x) \downarrow$.
- When the computation halts for some s, we simply write $\Phi_i^A(x) \downarrow$.
- If no value s exists, we write $\Phi_i^A(x) \uparrow$.
- We denote with $\phi_i^A(x)$ the **output** of the computation (defined iff $\Phi_i^A(x) \downarrow$)

Introduction

Notation

Let $\mathcal{M} = \{M_0, M_1, M_2, \ldots\}$ be the set of all Turing Machines working on $\Sigma = \{0, 1\}$.
We denote with $\Phi^A_{i,s}(x)$ the **computation** of M_i for $s \in \mathbb{N}$ steps on input $x \in \mathbb{N}$ while having access to an oracle for the subset $A \subseteq \mathbb{N}$

- If the computation halts after s steps, we write $\Phi^A_{i,s}(x) \downarrow$.
- When the computation halts for some s, we simply write $\Phi^A_i(x) \downarrow$.
- If no value s exists, we write $\Phi^A_i(x) \uparrow$.
- We denote with $\phi^A_i(x)$ the **output** of the computation (defined iff $\Phi^A_i(x) \downarrow$)

Given a set $S \subseteq \mathbb{N}$, we say that:

- S is **semi-decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in S$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$.
- S is **decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in \mathbb{N}$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$ if $x \in S$, otherwise $\phi_i(x) = 0$.
- S is **recursively enumerable** if there is an algorithmic procedure $\mathcal{A} : \mathbb{N} \to \{0, 1\}$ such that $S = \{A(0), A(1), A(2), \ldots\}$

Obs. 1: S is semi-decidable if and only if it is r.e.

Obs. 2: S is decidable if and only if both S and $\overline{S}$ are semi-decidable.

Introduction

Definitions

Given a set $S \subseteq \mathbb{N}$, we say that:

- S is **semi-decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in S$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$.
- S is **decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in \mathbb{N}$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$ if $x \in S$, otherwise $\phi_i(x) = 0$.
- S is **recursively enumerable** if there is an algorithmic procedure $\mathcal{A} : \mathbb{N} \to \{0, 1\}$ such that $S = \{A(0), A(1), A(2), \ldots\}$

Obs. 1: S is semi-decidable if and only if it is r.e.

Obs. 2: S is decidable if and only if both S and $\overline{S}$ are semi-decidable.

Introduction

Definitions

Given a set $S \subseteq \mathbb{N}$, we say that:

- S is **semi-decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in S$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$.
- S is **decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in \mathbb{N}$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$ if $x \in S$, otherwise $\phi_i(x) = 0$.
- S is **recursively enumerable** if there is an algorithmic procedure $\mathcal{A} : \mathbb{N} \to \{0, 1\}$ such that $S = \{A(0), A(1), A(2), \dots\}$

Obs. 1: S is semi-decidable if and only if it is r.e.

Obs. 2: S is decidable if and only if both S and $\overline{S}$ are semi-decidable.

Given a set $S \subseteq \mathbb{N}$, we say that:

- S is **semi-decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in S$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$.
- S is **decidable** if $\exists i \in \mathbb{N}$ such that $\forall x \in \mathbb{N}$ it holds that $\Phi_i(x) \downarrow$ and $\phi_i(x) = 1$ if $x \in S$, otherwise $\phi_i(x) = 0$.
- S is **recursively enumerable** if there is an algorithmic procedure $\mathcal{A} : \mathbb{N} \to \{0, 1\}$ such that $S = \{A(0), A(1), A(2), \dots\}$

Obs. 1: S is semi-decidable if and only if it is r.e.

Obs. 2: S is decidable if and only if both S and $\overline{S}$ are semi-decidable.

Introduction

Turing's work

- Turing [Tur37] proved that there are some problems that are some sets that are semi-decidable but undecidable (e.g. the set $H = \{(i, x) \mid \Phi_i(x) \downarrow\}$).
- He also proved that some sets cannot be semi-decided (e.g. the set $\overline{H}$).
- This gives three degrees of computability: **solvable** problems, **semi-solvable** problems and **unsolvable** problems.

Are there some other degrees of computability?

- Turing [Tur37] proved that there are some problems that are some sets that are semi-decidable but undecidable (e.g. the set $H = \{(i, x) \mid \Phi_i(x) \downarrow\}$).
- He also proved that some sets cannot be semi-decided (e.g. the set $\overline{H}$).
- This gives three degrees of computability: **solvable** problems, **semi-solvable** problems and **unsolvable** problems.

Are there some other degrees of computability?

Introduction

Turing's work

- Turing [Tur37] proved that there are some problems that are some sets that are semi-decidable but undecidable (e.g. the set $H = \{(i, x) \mid \Phi_i(x) \downarrow\}$).
- He also proved that some sets cannot be semi-decided (e.g. the set $\overline{H}$).
- This gives three degrees of computability: **solvable** problems, **semi-solvable** problems and **unsolvable** problems.

Are there some other degrees of computability?

Introduction

Turing's work

- Turing [Tur37] proved that there are some problems that are some sets that are semi-decidable but undecidable (e.g. the set $H = \{(i, x) \mid \Phi_i(x) \downarrow\}$).
- He also proved that some sets cannot be semi-decided (e.g. the set $\overline{H}$).
- This gives three degrees of computability: **solvable** problems, **semi-solvable** problems and **unsolvable** problems.

Are there some other degrees of computability?

Table of Contents

Degrees of Unsolvability

Turing degrees

Post [Pos44] formalized the idea of computability degrees through **Turing reductions**.

- Given $A, B \subseteq \mathbb{N}$, we say that A is Turing reducible to B, written as $A \leq_T B$, if $\exists i \in \mathbb{N}$ such that $\Phi_i^B(x)$ for all $x \in \mathbb{N}$ and $\phi_i^B = A$.
- We say that A and B are *Turing equivalent*, written as $A \equiv_T B$, when $A \leq_T B$ and $B \leq_T A$
- $\equiv_T$ is an equivalence relation over the set $2^{\mathbb{N}}$, inducing the quotient set $\mathcal{D} = 2^{\mathbb{N}}/_{\equiv_T}$. Each equivalence class of $\mathcal{D}$ is referred to as a **Turing degree**

Post [Pos44] formalized the idea of computability degrees through **Turing reductions**.

- Given $A, B \subseteq \mathbb{N}$, we say that A is Turing reducible to B, written as $A \leq_T B$, if $\exists i \in \mathbb{N}$ such that $\Phi_i^B(x)$ for all $x \in \mathbb{N}$ and $\phi_i^B = A$.
- We say that A and B are *Turing equivalent*, written as $A \equiv_T B$, when $A \leq_T B$ and $B \leq_T A$
- $\equiv_T$ is an equivalence relation over the set $2^{\mathbb{N}}$, inducing the quotient set $\mathcal{D} = 2^{\mathbb{N}}/_{\equiv_T}$. Each equivalence class of $\mathcal{D}$ is referred to as a **Turing degree**

Degrees of Unsolvability

Turing degrees

Post [Pos44] formalized the idea of computability degrees through **Turing reductions**.

- Given $A, B \subseteq \mathbb{N}$, we say that A is Turing reducible to B, written as $A \leq_T B$, if $\exists i \in \mathbb{N}$ such that $\Phi_i^B(x)$ for all $x \in \mathbb{N}$ and $\phi_i^B = A$.
- We say that A and B are *Turing equivalent*, written as $A \equiv_T B$, when $A \leq_T B$ and $B \leq_T A$
- $\equiv_T$ is an equivalence relation over the set $2^{\mathbb{N}}$, inducing the quotient set $\mathcal{D} = 2^{\mathbb{N}}/_{\equiv_T}$. Each equivalence class of $\mathcal{D}$ is referred to as a **Turing degree**

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.
$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1**: There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2**: 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.**: There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.
$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1**: There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2**: 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.**: There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.
$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1**: There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2**: 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.**: There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.
$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1:** There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2:** 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.:** There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.
$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1:** There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2:** 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.:** There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.
$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1**: There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2**: 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.**: There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.

$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1**: There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2**: 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.**: There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

Turing degrees

We say that $[A]$ is lower than $[B]$, written as $[A] \preceq [B]$, if $A \leq_T B$.
$\preceq$ is a (partial) order over the set $\mathcal{D}$, forming an hierarchy of unsolvability degree.

- **Prop. 1**: There is an unique degree containing all the decidable problems
  - Each decidable problem can be solved by ignoring the provided oracle
  - This unique class is referred to as the 0 degree (formally $0 = [\varnothing]$)
- **Prop. 2**: 0 is a minimal degree
  - If $[A] \preceq 0$ then A can be decided by reducing it to any decidable problem, thus $[A] = 0$
- **Obs.**: There are minimal degrees different from 0
  - E.g. the set $\overline{H}$

Degrees of Unsolvability

The jump operator

The strict relation $[A] \prec [B]$ between degrees can be easily forced through an operator known as the **Turing jump**.

- Given a set $X \subseteq \mathbb{N}$, the *Turing jump* of X is the set $X' = \{i \mid \Phi_i^X(i) \downarrow\}$
- It's easy to see that $X \not\leq_T X'$ since X' is obtained by forcing a variant of the Halting problem on TMs with X as an oracle
- In particular, the jump $0'$ of 0 is exactly the class containing the Halting problem, meaning that $0' = [H]$
- **Prop.**: $A \in 0'$ if and only if A is r.e.
  - If $A \in 0'$ then $A \leq_T H$, thus A is r.e. since H is r.e.
  - If A is r.e. then it has a semi-decider M_i. We can build a new semi-decider M_j such that $\Phi_j(x) \downarrow$ if $\phi_i(x) = 1$, otherwise $\Phi_j(x) \uparrow$. By construction, $x \in A$ iff $(j, x) \in H$.

Degrees of Unsolvability

The jump operator

The strict relation $[A] \prec [B]$ between degrees can be easily forced through an operator known as the **Turing jump**.

- Given a set $X \subseteq \mathbb{N}$, the *Turing jump* of X is the set $X' = \{i \mid \Phi_i^X(i) \downarrow\}$
- It's easy to see that $X \not\leq_T X'$ since X' is obtained by forcing a variant of the Halting problem on TMs with X as an oracle
- In particular, the jump $0'$ of 0 is exactly the class containing the Halting problem, meaning that $0' = [H]$
- **Prop.**: $A \in 0'$ if and only if A is r.e.
  - If $A \in 0'$ then $A \leq_T H$, thus A is r.e. since H is r.e.
  - If A is r.e. then it has a semi-decider M_i. We can build a new semi-decider M_j such that $\Phi_j(x) \downarrow$ if $\phi_i(x) = 1$, otherwise $\Phi_j(x) \uparrow$. By construction, $x \in A$ iff $(j, x) \in H$.

Degrees of Unsolvability

The jump operator

The strict relation $[A] \prec [B]$ between degrees can be easily forced through an operator known as the **Turing jump**.

- Given a set $X \subseteq \mathbb{N}$, the *Turing jump* of X is the set $X' = \{i \mid \Phi_i^X(i) \downarrow\}$
- It's easy to see that $X \not\leq_T X'$ since X' is obtained by forcing a variant of the Halting problem on TMs with X as an oracle
- In particular, the jump $0'$ of 0 is exactly the class containing the Halting problem, meaning that $0' = [H]$
- **Prop.**: $A \in 0'$ if and only if A is r.e.
  - If $A \in 0'$ then $A \leq_T H$, thus A is r.e. since H is r.e.
  - If A is r.e. then it has a semi-decider M_i. We can build a new semi-decider M_j such that $\Phi_j(x) \downarrow$ if $\phi_i(x) = 1$, otherwise $\Phi_j(x) \uparrow$. By construction, $x \in A$ iff $(j, x) \in H$.

Degrees of Unsolvability

The jump operator

The strict relation $[A] \prec [B]$ between degrees can be easily forced through an operator known as the **Turing jump**.

- Given a set $X \subseteq \mathbb{N}$, the *Turing jump* of X is the set $X' = \{i \mid \Phi_i^X(i) \downarrow\}$
- It's easy to see that $X \not\leq_T X'$ since X' is obtained by forcing a variant of the Halting problem on TMs with X as an oracle
- In particular, the jump $0'$ of 0 is exactly the class containing the Halting problem, meaning that $0' = [H]$
- **Prop.**: $A \in 0'$ if and only if A is r.e.
  - If $A \in 0'$ then $A \leq_T H$, thus A is r.e. since H is r.e.
  - If A is r.e. then it has a semi-decider M_i. We can build a new semi-decider M_j such that $\Phi_j(x) \downarrow$ if $\phi_i(x) = 1$, otherwise $\Phi_j(x) \uparrow$. By construction, $x \in A$ iff $(j, x) \in H$.

Degrees of Unsolvability

The jump operator

The strict relation $[A] \prec [B]$ between degrees can be easily forced through an operator known as the **Turing jump**.

- Given a set $X \subseteq \mathbb{N}$, the *Turing jump* of X is the set $X' = \{i \mid \Phi_i^X(i) \downarrow\}$
- It's easy to see that $X \not\leq_T X'$ since X' is obtained by forcing a variant of the Halting problem on TMs with X as an oracle
- In particular, the jump $0'$ of 0 is exactly the class containing the Halting problem, meaning that $0' = [H]$
- **Prop.**: $A \in 0'$ if and only if A is r.e.
  - If $A \in 0'$ then $A \leq_T H$, thus A is r.e. since H is r.e.
  - If A is r.e. then it has a semi-decider M_i. We can build a new semi-decider M_j such that $\Phi_j(x) \downarrow$ if $\phi_i(x) = 1$, otherwise $\Phi_j(x) \uparrow$. By construction, $x \in A$ iff $(j, x) \in H$.

Degrees of Unsolvability

The jump operator

The strict relation $[A] \prec [B]$ between degrees can be easily forced through an operator known as the **Turing jump**.

- Given a set $X \subseteq \mathbb{N}$, the *Turing jump* of X is the set $X' = \{i \mid \Phi_i^X(i) \downarrow\}$
- It's easy to see that $X \not\leq_T X'$ since X' is obtained by forcing a variant of the Halting problem on TMs with X as an oracle
- In particular, the jump $0'$ of 0 is exactly the class containing the Halting problem, meaning that $0' = [H]$
- **Prop.**: $A \in 0'$ if and only if A is r.e.
  - If $A \in 0'$ then $A \leq_T H$, thus A is r.e. since H is r.e.
  - If A is r.e. then it has a semi-decider M_i. We can build a new semi-decider M_j such that $\Phi_j(x) \downarrow$ if $\phi_i(x) = 1$, otherwise $\Phi_j(x) \uparrow$. By construction, $x \in A$ iff $(j, x) \in H$.

Degrees of Unsolvability

Post's problem

- The jump operator can be iteratively applied to get infinite levels of the hierarchy. This makes "going upwards" a less interesting question. This gave Post the idea of exploring the hierarchy sideways.
- Post proved that for each degree A there is another degree B that is incomparable with A.
  - **Obs.**: Simpson [Sim77] proved that the first-order theory of $\mathcal{D}$ over the language $(\preceq, =)$ is many-one equivalent to the theory of true second-order arithmetic.
- After exploring many results of this type, Post's work stopped on the following simple problem: is there a degree that lies between 0 and $0'$?
- This problem was only solved 12 years later independently by Friedberg [Fri57] and Muchnik [Muc56]

Degrees of Unsolvability

Post's problem

- The jump operator can be iteratively applied to get infinite levels of the hierarchy. This makes "going upwards" a less interesting question. This gave Post the idea of exploring the hierarchy sideways.
- Post proved that for each degree A there is another degree B that is incomparable with A.
  - **Obs.**: Simpson [Sim77] proved that the first-order theory of $\mathcal{D}$ over the language $(\preceq, =)$ is many-one equivalent to the theory of true second-order arithmetic.
- After exploring many results of this type, Post's work stopped on the following simple problem: is there a degree that lies between 0 and $0'$?
- This problem was only solved 12 years later independently by Friedberg [Fri57] and Muchnik [Muc56]

Degrees of Unsolvability

Post's problem

- The jump operator can be iteratively applied to get infinite levels of the hierarchy. This makes "going upwards" a less interesting question. This gave Post the idea of exploring the hierarchy sideways.
- Post proved that for each degree A there is another degree B that is incomparable with A.
  - **Obs.**: Simpson [Sim77] proved that the first-order theory of $\mathcal{D}$ over the language $(\preceq, =)$ is many-one equivalent to the theory of true second-order arithmetic.
- After exploring many results of this type, Post's work stopped on the following simple problem: is there a degree that lies between 0 and $0'$?
- This problem was only solved 12 years later independently by Friedberg [Fri57] and Muchnik [Muc56]

Degrees of Unsolvability

Post's problem

- The jump operator can be iteratively applied to get infinite levels of the hierarchy. This makes "going upwards" a less interesting question. This gave Post the idea of exploring the hierarchy sideways.
- Post proved that for each degree A there is another degree B that is incomparable with A.
  - **Obs.**: Simpson [Sim77] proved that the first-order theory of $\mathcal{D}$ over the language $(\preceq, =)$ is many-one equivalent to the theory of true second-order arithmetic.
- After exploring many results of this type, Post's work stopped on the following simple problem: is there a degree that lies between 0 and $0'$?
- This problem was only solved 12 years later independently by Friedberg [Fri57] and Muchnik [Muc56]

Degrees of Unsolvability

Post's problem

- The jump operator can be iteratively applied to get infinite levels of the hierarchy. This makes "going upwards" a less interesting question. This gave Post the idea of exploring the hierarchy sideways.
- Post proved that for each degree A there is another degree B that is incomparable with A.
  - **Obs.**: Simpson [Sim77] proved that the first-order theory of $\mathcal{D}$ over the language $(\preceq, =)$ is many-one equivalent to the theory of true second-order arithmetic.
- After exploring many results of this type, Post's work stopped on the following simple problem: is there a degree that lies between 0 and $0'$?
- This problem was only solved 12 years later independently by Friedberg [Fri57] and Muchnik [Muc56]

Bibliography

[Fri57] Richard M. Friedberg. “Two recursively enumerable sets of incomparable degrees of unsolvability (solution of Post’s problem, 1944)”. In: *Proceedings of the National Academy of Sciences* (1957).

[Muc56] Albert A. Muchnik. “On the unsolvability of the problem of reducibility in the theory of algorithms”. In: *Dokl. Akad. Nauk SSSR* (1956).

[Pos44] Emil L. Post. “Recursively enumerable sets of positive integers and their decision problems”. In: *Bull. Amer. Math. Soc.* (1944).

[Sim77] Stephen G. Simpson. “First-Order Theory of the Degrees of Recursive Unsolvability”. In: *Annals of Mathematics* (1977).

[Tur37] Alan M. Turing. “On Computable Numbers, with an Application to the Entscheidungsproblem”. In: *Proceedings of the London Mathematical Society* (1937).