Undecidability II

Lecture 12
Example of Undecidable Language

\[ \text{SELFREJECT} = \{ \langle M \rangle \mid M \text{ rejects } \langle M \rangle \} \]

\( M = \text{Turing Machine (piece of executable code)} \)

\( \langle M \rangle = \text{encoding of } M \text{ as a string (source code for } M) \)

\( \langle M \rangle \text{ is what you would feed to a universal TM,} \)

\( \text{that would allow it to simulate } M. \)

\( \text{(e.g. } TM \text{ that rejects everything.} \)

TM that rejects every description of a TM are in that language.
Showing Undecidability

To show \( L \) is undecidable, reduce some undecidable language to \( \mathcal{L} \).

\[
\mathcal{L} = \{ <M> | M \text{ halts on } <M> \}
\]

Claim: \( \mathcal{L} \) is undecidable

More general looking problem:

\[
\text{HALT} = \{ <M,w> | M \text{ halts on } w \}
\]

Claim: \( \text{HALT} \) is acceptable

The halting problem

Claim: \( \text{HALT} \) is undecidable
Claim: \( \text{HALT} \) is undecidable

Proof:
Suppose (towards contradiction) that there is a TM \( H \) that decides \( \text{HALT} \). Reduce from SELFHALT
NEVERACCEPT = \{ <M> | ACCEPT(M) = \emptyset \}

(is a TM useless or not?)

**Claim:** NEVERACCEPT is undecidable
How many Turing Machines?

• Fix a TM M and an input w.

• Build a new TM M’ with the following behavior:
  • M’ accepts its input iff M accepts w. (toss input out the window)

• Pseudocode:
  ```
  M'(x)
  Run M(w)
  ```
How many Turing Machines?

- Fix a TM $M$ and an input $w$.

- Build a new TM $M'$ with the following behavior:
  - $M'$ accepts its input iff $M$ accepts $w$. (toss input out the window)

- Pseudocode:

  $M'(x)$

  Run $M(w)$

  $M$
How many Turing Machines?

- Fix a TM $M$ and an input $w$.

- Build a new TM $M'$ with the following behavior:
  - $M'$ accepts its input iff $M$ accepts $w$. (toss input out the window)

- Pseudocode:
  
  $M'(x)$
  
  Run $M(w)$

$w$ hardcoded and $M$ hardcoded in $M'$
• Build M’?

Write a program

Input <M,w>: M - Turing Machine, w - string

Output <M’>: M’ - Turing Machine,

s.t. for any string x, M’ accepts x iff M accepts w.

• could produce M’ ourselves (write pseudocode).

So far, when we talk about reduction, WE are doing the reduction.
• Now, we need to describe how to do this transformation
  • by writing code that performs the transformation
NEVERACCEPT = \{ <M> \mid ACCEPT(M) = \emptyset \} \\
(M accepts nothing)

**Claim:** NEVERACCEPT is undecidable

We will assume we know the following:

\[ ACCEPT = \{ <M,w> \mid M \text{ accepts } w \} \text{ is undecidable} \]

**Proof:**

Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.
\[ \text{NEVERACCEPT} = \{ <M> \mid \text{ACCEPT}(M) = \emptyset \} \]

**Claim:** NEVERACCEPT is undecidable

**Proof:**

Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.
NEVERACCEPT = \{ <M> \mid ACCEPT(M) = \emptyset \}\\

Claim: NEVERACCEPT is undecidable

Proof:
Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.
NEVERACCEPT = \{ <M> | ACCEPT(M) = \emptyset \}

**Claim**: NEVERACCEPT is undecidable

**Proof**: Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.

how many TMs?
when I design a compiler for a piece of code, I can’t worry about the input that this code will be fed many many years from now. x and w not related!
\[
\text{NEVERACCEPT} = \{ \langle M \rangle \mid \text{ACCEPT}(M) = \emptyset \}
\]

**Claim**: A decides ACCEPT

- **Case 1**: \( M \) accepts \( w \).
NEVERACCEPT = \{ <M> | ACCEPT(M) = \emptyset \}

**Claim**: A decides ACCEPT

- **Case 1**: M accepts w.

  Implies M' accepts everything (by def. of M').

  Implies M' not in NEVERACCEPT (by def of NEVERACCEPT)

  Implies NA rejects <M'> (by def of NA)

  Implies A accepts <M,w> (by def of A)
NEVERACCEPT = \{ \langle M \rangle \mid \text{ACCEPT}(M) = \emptyset \} \\

**Claim**: A decides ACCEPT

- **Case 2**: M doesn’t accept w.
NEVERACCEPT = \{ <M> | ACCEPT(M) = \emptyset \} \\

**Claim:** A decides ACCEPT

- **Case 2:** M doesn’t accept w.

  Implies M’ doesn't accept anything (by def. of M’).

  Implies M’ in NEVERACCEPT (by def of NEVERACCEPT)

  Implies NA accepts <M’> (by def of NA)

  Implies A rejects <M, w> (by def of A)

These two cases are exhaustive and imply A decides ACCEPT, contradiction
\[ \text{NeverAccept} := \{ \langle M \rangle | \text{Accept}(M) = \emptyset \} \]
\[ \text{NeverReject} := \{ \langle M \rangle | \text{Reject}(M) = \emptyset \} \]
\[ \text{NeverHalt} := \{ \langle M \rangle | \text{Halt}(M) = \emptyset \} \]
\[ \text{NeverDiverge} := \{ \langle M \rangle | \text{Diverge}(M) = \emptyset \} \]
DIVERGERSAME = \{ \langle M_1 \rangle \langle M_2 \rangle \mid DIVERGE(M_1) = DIVERGE(M_2) \} \\
Claim: Undecidable
**Theorem 14.** The language $\text{DIVERGE}_{\text{SAME}} := \{ \langle M_1 \rangle \langle M_2 \rangle \mid \text{DIVERGE}(M_1) = \text{DIVERGE}(M_2) \}$ is undecidable.

**Proof:** Suppose for the sake of argument that there is a Turing machine $DS$ that decides $\text{DIVERGE}_{\text{SAME}}$. Then we can build a Turing machine $ND$ that decides $\text{NEVERDIVERGE}$ as follows. Fix a Turing machine $Y$ that accepts $\Sigma^*$ (for example, by defining $\delta(\text{start}, a) = (\text{accept}, \cdot, \cdot)$ for all $a \in \Gamma$). Given an arbitrary Turing machine encoding $\langle M \rangle$ as input, $ND$ writes the string $\langle M \rangle \langle Y \rangle$ onto the tape and then passes control to $DS$. There are two cases to consider:

- If $DS$ accepts $\langle M \rangle \langle Y \rangle$, then $\text{DIVERGE}(M) = \text{DIVERGE}(Y) = \emptyset$, so $\langle M \rangle \in \text{NEVERDIVERGE}$.
- If $DS$ rejects $\langle M \rangle \langle Y \rangle$, then $\text{DIVERGE}(M) \neq \text{DIVERGE}(Y) = \emptyset$, so $\langle M \rangle \notin \text{NEVERDIVERGE}$.

In short, $ND$ accepts $\langle M \rangle$ if and only if $\langle M \rangle \in \text{NEVERDIVERGE}$, which is impossible. We conclude that $DS$ does not exist. $\square$
Rice’s Theorem

- We want to answer questions of the form “does the language this machine accepts have some interesting property?”

- \( L = \{\text{set of acceptable languages that is not empty and is not the set of all languages}\} \)

  - e.g. \( L = \text{set of all languages containing the word “surfing”} \)

    - Define \( \text{ACCEPTIN}(L) = \{<M> | \text{ACCEPT}(M) \text{ is in } L\} \)

- \( L = \emptyset : \text{ACCEPTIN}(\emptyset) \) is decidable (always say no, no language is element of \( \emptyset \))

- \( L = \text{everything} : \text{ACCEPTIN}(\text{all}) \) is decidable (always say yes: does this TM accept a language?)

- For every other \( L \) \( \text{ACCEPTIN}(L) \) is undecidable
Rice’s Theorem

Rice’s Theorem. Let \( \mathcal{L} \) be any set of languages that satisfies the following conditions:

- There is a Turing machine \( Y \) such that \( \text{Accept}(Y) \in \mathcal{L} \).
- There is a Turing machine \( N \) such that \( \text{Accept}(N) \notin \mathcal{L} \).

The language \( \text{AcceptIn}(\mathcal{L}) := \{ \langle M \rangle \mid \text{Accept}(M) \in \mathcal{L} \} \) is undecidable.

To Show \( \text{AcceptIn}(\mathcal{L}) \) is undecidable

Reduce from \( \text{HALT} = \{ \langle M, w \rangle \mid M \text{ halts on } w \} \)
Rice’s Theorem

- $\text{ACCEPTIN}(L) = \{<M>|\text{ACCEPT}(M) \text{ is in } L\}$
- $\text{HALT} = \{<M,w>| M \text{ halts on } w\}$
Rice’s Theorem

- ACCEPTIN(L) = \{<M>|ACCEPT(M) is in L\}

\[HALT = \{<M,w>|M \text{ halts on } w\}\]

\[M \text{ halts on } W \text{ iff } ACCEPT(WTF) \text{ is in } L\]
Rice’s Theorem

- $\text{ACCEPTIN}(L) = \{ <M> | \text{ACCEPT}(M) \text{ is in } L \}$

$\text{HALT} = \{ <M,w> \mid M \text{ halts on } w \}$

$M$ halts on $w$ iff $\text{ACCEPT}(\text{WTF})$ is in $L$
**Rice’s Theorem**

- \( \text{ACCEPTIN}(L) = \{<M>|\text{ACCEPT}(M)\text{ is in } L\} \)

\[ \text{HALT} = \{ <M,w> | M \text{ halts on } w \} \]

\( M \text{ halts on } w \) iff \( \text{ACCEPT}(\text{WTF}) \) is in \( L \)

Assume \( \phi \) not in \( L \). Let \( Y \) be a TM so that \( \text{ACCEPT}(Y) \) in \( L \)
Rice’s Theorem

- \( \text{ACCEPTIN}(L) = \{<M>|\text{ACCEPT}(M) \text{ is in } L\} \)
Rice’s Theorem

- $\text{ACCEPTIN}(L) = \{<M>|\text{ACCEPT}(M) \text{ is in } L\}$

\[
\begin{align*}
\text{HALT} & \\
\text{WTF}(x) & \\
M(w) & \\
Y(x) & \\
\text{AIL} & \\
\end{align*}
\]

- if $M$ halts on $w$ then $\text{WTF}(x)$ is $Y(x)$ and $\text{ACCEPT}(\text{WTF}) = \text{ACCEPT}(Y)$ in $L$, AIL accepts
- if $M$ doesn't halt on $w$ then $\text{WTF}(x)$ never halts
  
  so $\text{ACCEPT}(\text{WTF}) = \emptyset$, not in $L$, AIL rejects

\[\phi \text{ not in } L\]
Rice’s Theorem

- ACCEPTIN(L) = \{<M>|\text{ACCEPT}(M) \text{ is in } L\}

H accepts <M,w> iff H halts on w!

\text{contradiction}
Rice’s Theorem

**Rice’s Theorem.** Let \( \mathcal{L} \) be any set of languages that satisfies the following conditions:
- There is a Turing machine \( Y \) such that \( \text{Accept}(Y) \in \mathcal{L} \).
- There is a Turing machine \( N \) such that \( \text{Accept}(N) \not\in \mathcal{L} \).

The language \( \text{AcceptIn}(\mathcal{L}) := \{ \langle M \rangle \mid \text{Accept}(M) \in \mathcal{L} \} \) is undecidable.

- example: \( \{ \langle M \rangle \mid M \text{ accepts the empty string} \} \)
Rice’s Theorem

Rice’s Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:

- There is a Turing machine $Y$ such that $\text{Accept}(Y) \in \mathcal{L}$.
- There is a Turing machine $N$ such that $\text{Accept}(N) \notin \mathcal{L}$.

The language $\text{AcceptIn}(\mathcal{L}) := \{ \langle M \rangle \mid \text{Accept}(M) \in \mathcal{L} \}$ is undecidable.

- example: $\{\langle M \rangle \mid M$ accepts the empty string$\}$

Let $\mathcal{L}$ be the set of all languages that contain the empty string. Then $\text{AcceptIn}(\mathcal{L}) = \{ \langle M \rangle \mid M$ accepts given an empty initial tape$\}$. 
Rice’s Theorem

Rice’s Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:

- There is a Turing machine $Y$ such that $\text{Accept}(Y) \in \mathcal{L}$.
- There is a Turing machine $N$ such that $\text{Accept}(N) \notin \mathcal{L}$.

The language $\text{AcceptIn}(\mathcal{L}) := \{\langle M \rangle \mid \text{Accept}(M) \in \mathcal{L}\}$ is undecidable.

- example: $\{\langle M \rangle \mid M$ accepts the empty string $\}$

Let $\mathcal{L}$ be the set of all languages that contain the empty string. Then $\text{AcceptIn}(\mathcal{L}) = \{\langle M \rangle \mid M$ accepts given an empty initial tape $\}$.

  - $M_1$ accepts nothing: empty string is not in $\emptyset$
  
  - $M_2$ accepts everything: empty string is in $S^*$
Rice’s Theorem

Rice’s Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:

- There is a Turing machine $Y$ such that $\text{Accept}(Y) \in \mathcal{L}$.
- There is a Turing machine $N$ such that $\text{Accept}(N) \notin \mathcal{L}$.

The language $\text{AcceptIN}(\mathcal{L}) := \{ \langle M \rangle \mid \text{Accept}(M) \in \mathcal{L} \}$ is undecidable.

example: $\{<M>| M \text{ accepts regular language}\}$
Rice’s Theorem

Rice’s Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:

- There is a Turing machine $Y$ such that $\text{Accept}(Y) \in \mathcal{L}$.
- There is a Turing machine $N$ such that $\text{Accept}(N) \notin \mathcal{L}$.

The language $\text{AcceptIn}(\mathcal{L}) := \{\langle M \rangle \mid \text{Accept}(M) \in \mathcal{L} \}$ is undecidable.

- example: $\{\langle M \rangle \mid M$ accepts the empty string$\}$

Let $\mathcal{L}$ be the set of all regular languages. Then $\text{AcceptIn}(\mathcal{L}) = \{\langle M \rangle \mid M$ accepts a regular language$\}$. 
Rice’s Theorem

Rice’s Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:
- There is a Turing machine $Y$ such that $\text{ACCEPT}(Y) \in \mathcal{L}$.
- There is a Turing machine $N$ such that $\text{ACCEPT}(N) \notin \mathcal{L}$.

The language $\text{ACCEPTIN}(\mathcal{L}) := \{ \langle M \rangle \mid \text{ACCEPT}(M) \in \mathcal{L} \}$ is undecidable.

- example: $\{ \langle M \rangle \mid M$ accepts the empty string $\}$

Let $\mathcal{L}$ be the set of all regular languages. Then $\text{ACCEPTIN}(\mathcal{L}) = \{ \langle M \rangle \mid M$ accepts a regular language $\}$.

- M1 accepts $O^*$
- M2 accepts $\{0^n1^n : n \geq 0\}$
Rice’s Rejection Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:
- There is a Turing machine $Y$ such that $\text{REJECT}(Y) \in \mathcal{L}$
- There is a Turing machine $N$ such that $\text{REJECT}(N) \notin \mathcal{L}$.

The language $\text{REJECT}IN(\mathcal{L}) := \{\langle M \rangle \mid \text{REJECT}(M) \in \mathcal{L}\}$ is undecidable.

Rice’s Halting Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:
- There is a Turing machine $Y$ such that $\text{HALT}(Y) \in \mathcal{L}$
- There is a Turing machine $N$ such that $\text{HALT}(N) \notin \mathcal{L}$.

The language $\text{HALT}IN(\mathcal{L}) := \{\langle M \rangle \mid \text{HALT}(M) \in \mathcal{L}\}$ is undecidable.

Rice’s Divergence Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:
- There is a Turing machine $Y$ such that $\text{DIVERGE}(Y) \in \mathcal{L}$
- There is a Turing machine $N$ such that $\text{DIVERGE}(N) \notin \mathcal{L}$.

The language $\text{DIVERGE}IN(\mathcal{L}) := \{\langle M \rangle \mid \text{DIVERGE}(M) \in \mathcal{L}\}$ is undecidable.

Rice’s Decision Theorem. Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:
- There is a Turing machine $Y$ such that $\text{decides}$ an language in $\mathcal{L}$.
- There is a Turing machine $N$ such that $\text{decides}$ an language not in $\mathcal{L}$.

The language $\text{DECIDE}IN(\mathcal{L}) := \{\langle M \rangle \mid M \text{ decides a language in } \mathcal{L}\}$ is undecidable.
Exercise:

The language $L := \{ \langle M, w \rangle \mid M \text{ accepts } w^k \text{ for every integer } k \geq 0 \}$ is undecidable.