Lecture 14:
• Undecidability
• Reductions

Reading:
Sipser Ch 4.2, 5.1

Mark Bun
October 26, 2021

https://forms.gle/LMB5MR8hSc5mxVt4A
Where we are and where we’re going

Church-Turing thesis: TMs capture all algorithms
Consequence: studying the limits of TMs reveals the limits of computation

\[
\{ \text{TM deciders} \} \text{ is countable}\\
\& \text{languages over } \Sigma \text{ is uncountable}
\]

Last time: Countability, uncountability, and diagonalization
Existential proof that there are undecidable and unrecognizable languages

Today: An explicit undecidable language
Reductions: Relate decidability / undecidability of different problems
An Explicit Undecidable Language
Last time:

Theorem: Let $X$ be any set. Then the power set $P(X)$ does not have the same size as $X$.

1) Assume, for the sake of contradiction, that there is a bijection $f: X \rightarrow P(X)$

2) “Flip the diagonal” to construct a set $S \in P(X)$ such that $f(x) \neq S$ for every $x \in X$

3) Conclude that $f$ is not onto, contradicting assumption that $f$ is a bijection
Specializing the proof

**Theorem**: Let $X$ be the set of all TM deciders. Then there exists an undecidable language in $P(\{0, 1\}^*)$

1) Assume, for the sake of contradiction, that $L: X \rightarrow P(\{0, 1\}^*)$ is onto \\
   Mapping from TM to the language it recognizes

2) “Flip the diagonal” to construct a language $UD \in P(\{0, 1\}^*)$ such that $L(M) \neq UD$ for every $M \in X$

3) Conclude that $L$ is not onto, a contradiction
An explicit undecidable language

<table>
<thead>
<tr>
<th>TM $M$</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_2$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_3$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_4$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\vdots$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Why is it possible to enumerate all TMs like this?

a) The set of all TMs is finite
b) The set of all TMs is countably infinite
[c) The set of all TMs is uncountable
An explicit undecidable language

<table>
<thead>
<tr>
<th>TM $M$</th>
<th>$M(\langle M_1 \rangle)$?</th>
<th>$M(\langle M_2 \rangle)$?</th>
<th>$M(\langle M_3 \rangle)$?</th>
<th>$M(\langle M_4 \rangle)$?</th>
<th>$D(\langle D \rangle)$?</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>Y N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>...</td>
</tr>
<tr>
<td>$M_2$</td>
<td>N</td>
<td>N Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>$M_3$</td>
<td>Y</td>
<td>Y</td>
<td>Y N</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>$M_4$</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>$D$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y N</td>
</tr>
</tbody>
</table>

$UD = \{ \langle M \rangle \mid M \text{ is a TM that does not accept on input } \langle M \rangle \} $

Claim: $UD$ is undecidable

Assume for contradiction $\exists \text{ TM } D$ deciding $UD$

Case 1: If $D$ accepts $\langle 0 \rangle$, then by definition of $UD$, $\langle 0 \rangle \not\in UD$ ✗

Case 2: If $D$ does not accept $\langle 0 \rangle$, then by definition of $UD$, $\langle 0 \rangle \in UD$ ✗
An explicit undecidable language

Theorem: \( UD = \{ \langle M \rangle \mid M \text{ is a TM that does not accept on input } \langle M \rangle \} \) is undecidable

Proof: Suppose for contradiction, that TM \( D \) decides \( UD \)

Either:

1) \( D \) accepts \( \langle D \rangle \) \( \implies \langle D \rangle \notin UD \) (by def. of \( UD \))
   \( \implies D \) does the wrong thing on input \( \langle D \rangle \)

2) \( D \) does not accept \( \langle D \rangle \) \( \implies \langle D \rangle \in UD \) (by def. of \( UD \))
   \( \implies D \) does the wrong thing.
A more useful undecidable language

\[ A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM that accepts input } w \} \]

**Theorem:** \( A_{TM} \) is undecidable

**Proof:** Assume for the sake of contradiction that TM \( H \) decides \( A_{TM} \):

\[
H(\langle M, w \rangle) = \begin{cases} 
\text{accept} & \text{if } M \text{ accepts } w \\
\text{reject} & \text{if } M \text{ does not accept } w
\end{cases}
\]

\( \text{(either } M \text{ rejects } w \text{ or } M \text{ loops on } w) \)

**Idea:** Show that \( H \) can be used to construct a decider for the (undecidable) language \( UD \) -- a contradiction.

"Reduction"
A more useful undecidable language

\[ UD = \{ \langle M \rangle \mid \text{TM } M \text{ does not accept on input } \langle M \rangle \} \]

\[ A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM that accepts input } w \} \]

Proof (continued):

Suppose, for contradiction, that \( H \) decides \( A_{TM} \)

Consider the following TM \( D \):

"On input \( \langle M \rangle \) where \( M \) is a TM:

1. Run \( H \) on input \( \langle M, \langle M \rangle \rangle \)
2. If \( H \) accepts, reject. If \( H \) rejects, accept."

Claim: \( D \) decides \( UD = \{ \langle M \rangle \mid \text{TM } M \text{ does not accept } \langle M \rangle \} \)

Case 1: If \( \langle M \rangle \in UD \Rightarrow M \text{ does not accept } \langle M \rangle \Rightarrow \langle M, \langle M \rangle \rangle \notin A_{TM} \Rightarrow H \text{ rejects} \Rightarrow D \text{ accepts}

Case 2: If \( \langle M \rangle \notin UD \Rightarrow M \text{ accepts } \langle M \rangle \Rightarrow \langle M, \langle M \rangle \rangle \in A_{TM} \Rightarrow H \text{ accepts} \Rightarrow D \text{ rejects}

...but this language is undecidable

10/26/2021
Unrecognizable Languages

Theorem: A language $L$ is decidable if and only if $L$ and $\overline{L}$ are both Turing-recognizable.

Proof: $\Rightarrow$

$L$ is decidable $\Rightarrow L$ is recognizable

$L$ is decidable $\Rightarrow \overline{L}$ is decidable (closure of decidable langs. under complement)

$\Rightarrow \overline{L}$ is recognizable

Application: $A_{TM}$ is "co-unrecognizable" meaning $A_{TM}$ is unrecognizable.

Proof: By Thm, $L$ is undecidable $\iff$ at least one of $L, \overline{L}$ unrecognizable.

$A_{TM}$ undecidable $\Rightarrow$ either $A_{TM}$ or $\overline{A_{TM}}$ unrecognizable

$\Rightarrow A_{TM}$ unrecognizable
Unrecognizable Languages

**Theorem:** A language $L$ is decidable if and only if $L$ and $\overline{L}$ are both Turing-recognizable.

**Proof:**

$\leq$ Suppose $L$ is recognized by TM $M$ and $\overline{L}$ is recognized by TM $N$.

**Goal:** Construct a decider $V$ for $L$ (using $M$ and $N$).

$V$: On input $w$:

Repeat the following forever:

1. Run $M$ for one step on $w$.
2. Run $N$ for one step on $w$.
3. If $M$ accepts, accept; if $N$ accepts, reject.
Classes of Languages

- **Regular**
- **Decidable**
- **Recognizable**

- $A_{TM}$
- $\overline{A_{TM}}$

Languages:
- $\exists 0^n | n \geq 0^3$
- $\exists 0^n 1^n | n \geq 0^3$
Reductions
Scientists vs. Engineers

A computer scientist and an engineer are stranded on a desert island. They find two palm trees with one coconut on each. The engineer climbs a tree, picks a coconut and eats.

The computer scientist climbs the second tree, picks a coconut, climbs down, climbs up the first tree and places it there, declaring success.

“Now we’ve reduced the problem to one we’ve already solved.” (Please laugh)
Reductions

A reduction from problem \( A \) to problem \( B \) is an algorithm for problem \( A \) which uses an algorithm for problem \( B \) as a subroutine.

If such a reduction exists, we say “\( A \) reduces to \( B \)”
Reductions

A reduction from problem $A$ to problem $B$ is an algorithm for problem $A$ which uses an algorithm for problem $B$ as a subroutine.

If such a reduction exists, we say “$A$ reduces to $B$”

If $A$ reduces to $B$, and $B$ is decidable, what can we say about $A$?

a) $A$ is decidable
b) $A$ is undecidable
c) $A$ might be either decidable or undecidable
Two uses of reductions

Positive uses: If $A$ reduces to $B$ and $B$ is decidable, then $A$ is also decidable

$E_{DFA} = \{ \langle A \rangle \mid A \text{ is a DFA, } \lambda(A) = \emptyset \}$

$EQ_{DFA} = \{ \langle D_1, D_2 \rangle \mid D_1, D_2 \text{ are DFAs and } L(D_1) = L(D_2) \}$

Theorem: $EQ_{DFA}$ is decidable

Proof: The following TM decides $EQ_{DFA}$

On input $\langle D_1, D_2 \rangle$, where $\langle D_1, D_2 \rangle$ are DFAs:

1. Construct a DFA $D$ that recognizes the symmetric difference $L(D_1) \Delta L(D_2)$

2. Run the decider for $E_{DFA}$ on $\langle D \rangle$ and return its output
Two uses of reductions

Negative uses: If $A$ reduces to $B$ and $A$ is undecidable, then $B$ is also undecidable

$A_{TM} = \{\langle M, w \rangle \mid M \text{ is a TM that accepts input } w \}$

Suppose $H$ decides $A_{TM}$

Consider the following TM $D$.

On input $\langle M \rangle$ where $M$ is a TM:
1. Run $H$ on input $\langle M, \langle M \rangle \rangle$
2. If $H$ accepts, reject. If $H$ rejects, accept.

Claim: $D$ decides

$UD = \{\langle M \rangle \mid M \text{ is a TM that does not accept input } \langle M \rangle \}$
Two uses of reductions

Negative uses: If $A$ reduces to $B$ and $A$ is undecidable, then $B$ is also undecidable

Template for undecidability proof by reduction:
1. Suppose to the contrary that $B$ is decidable
2. Using a decider for $B$ as a subroutine, construct an algorithm deciding $A$
3. But $A$ is undecidable. Contradiction!
Halting Problem

Computational problem: Given a program (TM) and input $w$, does that program halt (either accept or reject) on input $w$?

Formulation as a language:

$$HALT_{TM} = \{\langle M, w \rangle \mid M \text{ is a TM that halts on input } w \}$$

Ex. $M =$ “On input $x$ (a natural number written in binary):

For each $y = 1, 2, 3, \ldots$:

If $y^2 = x$, accept. Else, continue.”

Is $\langle M, 101 \rangle \in HALT_{TM}$?

a) Yes, because $M$ accepts on input 101
b) Yes, because $M$ rejects on input 101
c) No, because $M$ rejects on input 101
d) No, because $M$ loops on input 101
Halting Problem

Computational problem: Given a program (TM) and input $w$, does that program halt (either accept or reject) on input $w$?

Formulation as a language:
$$HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM that halts on input } w \}$$

Ex. $M = \text{“On input } x \text{ (a natural number in binary):} \newline \text{For each } y = 1, 2, 3, \ldots : \newline \quad \text{If } y^2 = x, \text{ accept. Else, continue.”}$

$M' = \text{“On input } x \text{ (a natural number in binary):} \newline \text{For each } y = 1, 2, 3, \ldots, x : \newline \quad \text{If } y^2 = x, \text{ accept. Else, continue.} \newline \quad \text{Reject.”}$
Halting Problem

\[ HALT_{\text{TM}} = \{ \langle M, w \rangle \mid M \text{ is a TM that halts on input } w \} \]

Theorem: \( HALT_{\text{TM}} \) is undecidable

Proof: Suppose for contradiction that there exists a decider \( H \) for \( HALT_{\text{TM}} \). We construct a decider for \( V \) for \( A_{\text{TM}} \) as follows:

On input \( \langle M, w \rangle \): Input to \( A_{\text{TM}} \)
1. Run \( H \) on input \( \langle M, w \rangle \)
2. If \( H \) rejects, reject
3. If \( H \) accepts, run \( M \) on \( w \)
4. If \( M \) accepts, accept
Otherwise, reject.

This is a reduction from \( A_{\text{TM}} \) to \( HALT_{\text{TM}} \)