Undecidability and Reductions

CSCI 3130 Formal Languages and Automata Theory

Siu On CHAN
Fall 2020

Chinese University of Hong Kong
Undecidability

\[ A_{TM} = \{ \langle M, w \rangle \mid \text{Turing machine } M \text{ accepts input } w \} \]

Turing’s Theorem
The language \( A_{TM} \) is undecidable

Note: a Turing machine \( M \) may take as input its own description \( \langle M \rangle \)
Turing’s Theorem: Proof sketch (in Python)

Suppose function $H(M)$ correctly decides whether program $M$ halts, given its source code $\langle M \rangle$

```python
>>> M = "x = 1"
```

```python
>>> print(H(M))
True
```

```python
>>> M = ""
```

```python
>>> while True: continue
""
```

```python
>>> print(H(M))
False
```

$D$ checks whether itself halts using $H$ and does the opposite

```python
def D():
    if H(D):
        loop_forever()
```

Does $D$ halt?
Proof by contradiction:

Suppose $A_{TM}$ is decidable, then some TM $H$ decides $A_{TM}$:

$\langle M, w \rangle \rightarrow H$:
- accept if $M$ accepts $w$
- reject if $M$ rejects or loops on $w$
Proof by contradiction:

Suppose $A_{TM}$ is decidable, then some TM $H$ decides $A_{TM}$:

$$\langle M, w \rangle \rightarrow \begin{cases} 
\text{accept if } M \text{ accepts } w \\
\text{reject if } M \text{ rejects or loops on } w 
\end{cases}$$

Construct a new TM $D$ (that uses $H$ as a subroutine):

**Turing machine $D$: On input $\langle M \rangle$**

1. Run $H$ on input $\langle M, \langle M \rangle \rangle$
2. Output the opposite of $H$: If $H$ accepts, reject; if $H$ rejects, accept
Formal proof of Turing’s Theorem

What happens when $M = D$?

- $\langle M \rangle \rightarrow D$
  - accept if $M$ rejects or loops on $\langle M \rangle$
  - reject if $M$ accepts $\langle M \rangle$

- $\langle D \rangle \rightarrow D$
  - accept if $D$ rejects or loops on $\langle D \rangle$
  - reject if $D$ accepts $\langle D \rangle$
Formal proof of Turing’s Theorem

What happens when $M = D$?

$H$ never loops indefinitely, neither does $D$

If $D$ rejects $\langle D \rangle$, then $D$ accepts $\langle D \rangle$

If $D$ accepts $\langle D \rangle$, then $D$ rejects $\langle D \rangle$

Contradiction! $D$ cannot exist! $H$ cannot exist!
Proof of Turing’s theorem: conclusion

Proof by contradiction

Assume $A_{TM}$ is decidable
Then there are TM $H, H'$ and $D$
But $D$ cannot exist!

Conclusion

The language $A_{TM}$ is undecidable
Write an infinite table for the pairs \((M, w)\)

(Entries in this table are all made up for illustration)
## Diagonalization

<table>
<thead>
<tr>
<th>all possible Turing machines</th>
<th>inputs $w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>acc</td>
</tr>
<tr>
<td>$M_2$</td>
<td>rej</td>
</tr>
<tr>
<td>$M_3$</td>
<td>loop</td>
</tr>
<tr>
<td>$M_4$</td>
<td>acc</td>
</tr>
</tbody>
</table>

... 

Only look at those $w$ that *describe* Turing machines.
Diagonalization

If $A_{TM}$ is decidable, then TM $D$ is in the table.

<table>
<thead>
<tr>
<th>Turing machines</th>
<th>inputs $w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>acc</td>
</tr>
<tr>
<td></td>
<td>loop</td>
</tr>
<tr>
<td>$M_2$</td>
<td>rej</td>
</tr>
<tr>
<td></td>
<td>rej</td>
</tr>
<tr>
<td>$M_3$</td>
<td>loop</td>
</tr>
<tr>
<td></td>
<td>acc</td>
</tr>
<tr>
<td>$D$</td>
<td>rej</td>
</tr>
<tr>
<td></td>
<td>acc</td>
</tr>
<tr>
<td></td>
<td>rej</td>
</tr>
</tbody>
</table>

$\langle M_1 \rangle$, $\langle M_2 \rangle$, $\langle M_3 \rangle$, $\langle M_4 \rangle$, ...
## Diagonalization

<table>
<thead>
<tr>
<th>Turing machines</th>
<th>inputs $w$</th>
<th>$\langle M_1 \rangle$</th>
<th>$\langle M_2 \rangle$</th>
<th>$\langle M_3 \rangle$</th>
<th>$\langle M_4 \rangle$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>acc</td>
<td>loop</td>
<td>rej</td>
<td>rej</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_2$</td>
<td>rej</td>
<td>rej</td>
<td>acc</td>
<td>rej</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>$M_3$</td>
<td>loop</td>
<td>acc</td>
<td>acc</td>
<td>acc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D$</td>
<td>rej</td>
<td>acc</td>
<td>rej</td>
<td>rej</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

All possible

$D$ does the opposite of the diagonal entries

$D$ on $\langle M_i \rangle$ = opposite of $M_i$ on $\langle M_i \rangle$

$\langle D \rangle$ → accept if $D$ rejects or loops on $\langle D \rangle$

$\langle D \rangle$ → reject if $D$ accepts $\langle D \rangle$
We run into trouble when we look at \((D, \langle D \rangle)\)
Unrecognizable languages

The language $A_{TM}$ is recognizable but not decidable

How about languages that are not recognizable?

$\overline{A_{TM}} = \{ \langle M, w \rangle | M \text{ is a TM that does not accept } w \}$

$\overline{A_{TM}} = \{ \langle M, w \rangle | M \text{ rejects or loops on input } w \}$

Claim

The language $\overline{A_{TM}}$ is not recognizable
Theorem

If \( L \) and \( \overline{L} \) are both recognizable, then \( L \) is decidable

Proof of Claim from Theorem:

We know \( A_{TM} \) is recognizable

if \( \overline{A_{TM}} \) were also, then \( A_{TM} \) would be decidable

But Turing’s Theorem says \( A_{TM} \) is not decidable
Unrecognizable languages

Theorem

If \( L \) and \( \overline{L} \) are both recognizable, then \( L \) is decidable

Proof idea (flawed):

Let \( M = \text{TM recognizing } L \), \( M' = \text{TM recognizing } \overline{L} \)

The following Turing machine \( N \) decides \( L \):

Turing machine \( N \): On input \( w \)

1. Simulate \( M \) on input \( w \). If \( M \) accepts, accept
2. Simulate \( M' \) on input \( w \). If \( M' \) accepts, reject
Theorem

If $L$ and $\overline{L}$ are both recognizable, then $L$ is decidable

Proof idea (flawed):

Let $M = \text{TM recognizing } L$, $M' = \text{TM recognizing } \overline{L}$

The following Turing machine $N$ decides $L$:

Turing machine $N$: On input $w$

1. Simulate $M$ on input $w$. If $M$ accepts, accept
2. Simulate $M'$ on input $w$. If $M'$ accepts, reject

Problem: If $M$ loops on $w$, we will never go to step 2
Theorem

If $L$ and $\overline{L}$ are both recognizable, then $L$ is decidable

Proof idea (2nd attempt):

Let $M = \text{TM recognizing } L$, $M' = \text{TM recognizing } \overline{L}$

The following Turing machine $N$ decides $L$:

Turing machine $N$: On input $w$

For $t = 0, 1, 2, 3, \ldots$

Simulate first $t$ transitions of $M$ on input $w$.
If $M$ accepts, accept
Simulate first $t$ transitions of $M'$ on input $w$.
If $M'$ accepts, reject
Reductions
Reductions

Program $S$ reduces to Program $R$ solves Problem $B$ reduces to Problem $A$

Reducing $B$ to $A$

Transform program $R$ that solves $A$ into program $S$ that solves $B$

If you can reduce $B$ to $A$

Then you can solve problem $B$ using subroutine $R$ as a blackbox

Example from Lecture 16:

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

$A_{\text{NFA}} = \{ \langle N, w \rangle \mid N \text{ is an NFA that accepts input } w \}$

$A_{\text{NFA}}$ reduces to $A_{\text{DFA}}$ (by converting NFA into DFA)
Reductions in this course

If language $B$ reduces to language $A$, and $B$ is undecidable then $A$ is also undecidable.

Steps for showing a language $A$ to be undecidable:

1. If some TM $R$ decides $A$
2. Using $R$, build another TM $S$ that decides $B = A_{TM}$

But by Turing’s theorem, $A_{TM}$ is not decidable.
Another undecidable language

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

We’ll show:

HALT\textsubscript{TM} is an undecidable language

We will argue that

If HALT\textsubscript{TM} is decidable, then so is \( A_{TM} \)
Undecidability of halting

If \( \text{HALT}_{\text{TM}} \) can be decided, so can \( A_{\text{TM}} \)

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

Suppose \( \text{HALT}_{\text{TM}} \) is decidable by a Turing machine \( H \)

Then the following TM \( S \) decides \( A_{\text{TM}} \)

**Turing machine \( S \): On input \( \langle M, w \rangle \)**

1. Run \( H \) on input \( \langle M, w \rangle \)
2. If \( H \) rejects, reject
3. If \( H \) accepts, run the universal TM \( U \) on input \( \langle M, w \rangle \)
   - If \( U \) accepts, accept; else reject
Example 1

\[ A'_{TM} = \{ \langle M \rangle \mid M \text{ is a TM that accepts input } \varepsilon \} \]

Is \( A'_{TM} \) decidable? Why?

Undecidable!

Intuitive reason: To know whether \( M \) accepts \( \varepsilon \) seems to require simulating \( M \). But then we need to know whether \( M \) halts.
Example 1

\[ A'_{TM} = \{ \langle M \rangle \mid M \text{ is a TM that accepts input } \varepsilon \} \]

Is \( A'_{TM} \) decidable? Why?

Undecidable!

Intuitive reason:

To know whether \( M \) accepts \( \varepsilon \) seems to require simulating \( M \)

But then we need to know whether \( M \) halts

Let’s justify this intuition
Example 1: Figuring out the reduction

Suppose $A'_\text{TM}$ can be decided by a TM $R$

$\langle M' \rangle \rightarrow \text{accept if } M' \text{ accepts } \varepsilon$

$\langle M' \rangle \rightarrow \text{reject otherwise}$

We want to build a TM $S$

$\langle M, w \rangle \rightarrow \text{?}$

$\langle M' \rangle \rightarrow R \rightarrow \text{accept if } M \text{ accepts } w$

$\langle M' \rangle \rightarrow R \rightarrow \text{reject otherwise}$

$M'$ should be a Turing machine such that

outcome of $M'$ on input $\varepsilon = \text{outcome of } M \text{ on input } w$
Example 1: Implementing the reduction

\[ (M, w) \rightarrow ? \rightarrow (M') \]

\(M'\) should be a Turing machine such that

\(M'\) on input \(\varepsilon = M\) on input \(w\)

**Turing machine \(M'\):** On input \(z\)

1. Simulate \(M\) on input \(w\)
2. If \(M\) accepts \(w\), accept
3. If \(M\) rejects \(w\), reject

- If \(M\) accepts \(w\), \(M'\) accepts \(\varepsilon\)
- If \(M\) rejects \(w\), \(M'\) rejects \(\varepsilon\)
- If \(M\) loops on \(w\), \(M'\) loops on \(\varepsilon\)
Turing machine $S$: On input $\langle M, w \rangle$ where $M$ is a TM

1. Construct the following TM $M'$:

   $\langle M', w \rangle$ such that on input $z$,

   Simulate $M$ on input $w$ and accept/reject according to $M$

2. Run $R$ on input $\langle M' \rangle$ and accept/reject according to $R$
Example 1: The formal proof

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

Suppose \( A'_{\text{TM}} \) is decidable by a TM \( R \)
Consider the TM \( S \):

**TM \( S \):** On input \( \langle M, w \rangle \) where \( M \) is a TM

1. Construct the following TM \( M' \):

\[ M' = \text{a TM such that on input } z, \]

   Simulate \( M \) on input \( w \) and accept/reject according to \( M \)

2. Run \( R \) on input \( \langle M' \rangle \) and accept/reject according to \( R \)

Then \( S \) accepts \( \langle M, w \rangle \) if and only if \( M \) accepts \( w \)
So \( S \) decides \( A_{\text{TM}} \), which is impossible
Example 2

\[ A''_{TM} = \{ \langle M \rangle \mid M \text{ is a TM that accepts some input strings} \} \]

Is \( A''_{TM} \) decidable? Why?

Undecidable!

Intuitive reason:

To know whether \( M \) accepts some strings seems to require simulating \( M \)

But then we need to know whether \( M \) halts

Let’s justify this intuition
Suppose $A''_{TM}$ can be decided by a TM $R$

We want to build a TM $S$

$M'$ should be a Turing machine such that $M'$ accepts some strings if and only if $M$ accepts input $w$
Implementing the reduction

**Task:** Given \( \langle M, w \rangle \), construct \( M' \) so that

If \( M \) accepts \( w \), then \( M' \) accepts some input

If \( M \) does not accept \( w \), then \( M' \) accepts no inputs

**TM \( M' \): On input \( z \)**

1. Simulate \( M \) on input \( w \)
2. If \( M \) accepts, accept
3. Otherwise, reject
Example 2: The formal proof

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

Suppose \( A''_{TM} \) is decidable by a TM \( R \)

Consider the TM \( S \):

**TM \( S \):** On input \( \langle M, w \rangle \) where \( M \) is a TM

1. Construct the following TM \( M' \):
   \[ M' = \text{a TM such that on input } z, \]
   Simulate \( M \) on input \( w \) and accept/reject according to \( M \)
2. Run \( R \) on input \( \langle M' \rangle \) and accept/reject according to \( R \)

Then \( S \) accepts \( \langle M, w \rangle \) if and only if \( M \) accepts \( w \)

So \( S \) decides \( A_{TM} \), which is impossible
$E_{TM} = \{\langle M \rangle \mid M \text{ is a TM that accepts no input}\}$

Is $E_{TM}$ decidable?

Undecidable! We will show:

If $E_{TM}$ can be decided by some TM $R$

Then $A''_{TM}$ can be decided by another TM $S$

$A''_{TM} = \{\langle M \rangle \mid M \text{ is a TM that accepts some input strings}\}$
Example 3

\[ E_{TM} = \{ \langle M \rangle \mid M \text{ is a TM that accepts no input} \} \]
\[ A''_{TM} = \{ \langle M \rangle \mid M \text{ is a TM that accepts some input} \} \]

Then \( E_{TM} = \overline{A''_{TM}} \) (except ill-formatted strings, which we will ignore)

Suppose \( E_{TM} \) can be decided by some TM \( R \)

Consider the following Turing machine \( S \):

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

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

Then \( S \) decides \( \overline{A''_{TM}} \), a contradiction
EQ_{TM} = \{\langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs such that } L(M_1) = L(M_2) \}\n
Is EQ_{TM} decidable?

Undecidable!

We will show that EQ_{TM} can be decided by some TM R then EQ_{TM} can be decided by another TM S
Example 4: Setting up the reduction

\[ \text{EQ}_{TM} = \{ \langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs such that } L(M_1) = L(M_2) \} \]
\[ \text{E}_{TM} = \{ \langle M \rangle \mid M \text{ is a TM that accepts no input} \} \]

Given \( \langle M \rangle \), we need to construct \( \langle M_1, M_2 \rangle \) so that

- If \( M \) accepts no input, then \( M_1 \) and \( M_2 \) accept the same set of inputs
- If \( M \) accepts some input, then \( M_1 \) and \( M_2 \) do not accept the same set of inputs

Idea: Make \( M_1 = M \)

Make \( M_2 \) accept nothing
Example 4: The formal proof

\[ \text{EQ}_{\text{TM}} = \{ \langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs such that } L(M_1) = L(M_2) \} \]
\[ \text{ETM} = \{ \langle M \rangle \mid M \text{ is a TM that accepts no input} \} \]

Suppose \( \text{EQ}_{\text{TM}} \) is decidable and \( R \) decides it.

Consider the following Turing machine \( S \):

1. Construct a TM \( M_2 \) that rejects every input \( z \)
2. Run \( R \) on input \( \langle M, M_2 \rangle \) and accept/reject according to \( R \)

Then \( S \) accepts \( \langle M \rangle \) if and only if \( M \) accepts no input.

So \( S \) decides \( \text{ETM} \) which is impossible.