Undecidable Languages

✦ **The Question:** Are there languages that are not decidable by any Turing machine (TM)?
  ➔ i.e. Are there problems that cannot be solved by any algorithm?

✦ Consider the language:
  \[ A_{TM} = \{ <M,w> \mid M \text{ is a TM and } M \text{ accepts } w \} \]
  ➔ NOTE: \(<A,B,…>\) is just a string encoding the objects \(A, B, \ldots\)
  ➔ In particular, \(<M,w>\) is a string listing all the components of TM \(M\) (separated by #, for example) followed by the string \(w\)
  ➔ Given input \(<M,w>\), it should be easy to extract the info about \(M\) and to simulate \(M\) on \(w\) (try writing a TM to do this!)

✦ What can we say about \(A_{TM}\)?

\[ A_{TM} \text{ is Turing-recognizable} \]

✦ \(A_{TM}\) is Turing-recognizable: Recognizer TM \(U\) for \(A_{TM}\):

- On input string \(<M,w>\):
  - Simulate \(M\) on \(w\).
  - ACCEPT \(<M,w>\) if \(M\) halts & accepts \(w\); REJECT \(<M,w>\) if \(M\) halts & rejects (Loop (& thus reject \(<M,w>\)) if \(M\) ends up looping).
- \(U\) accepts \(<M,w>\) iff \(M\) accepts \(w\), i.e. \(L(U) = A_{TM}\)

Yeah, but is it decidable?!!
Is $A_{TM}$ decidable?

✦ No! $A_{TM} = \{ <M,w> \mid M$ is a TM and $M$ accepts $w \}$ is
undecidable! 1-slide Proof (by Contradiction):
1. Assume $A_{TM}$ is decidable $\Rightarrow$ there’s a decider $H$, $L(H) = A_{TM}$
2. $H$ on $<M,w> = ACC$ if $M$ accepts $w$
   $\quad$ REJ if $M$ rejects $w$ (halts in $q_{REJ}$ or loops on $w$)
3. Construct new TM $D$: On input $<M>$:
   Simulate $H$ on $<M,<M>>$ (here, $w = <M>$)
   If $H$ accepts, then REJ input $<M>
   If $H$ rejects, then ACC input $<M>
4. What happens when $D$ gets $<D>$ as input?
   D rejects $<D>$ if $H$ accepts $<D,<D>>$ if $D$ accepts $<D>$
   D accepts $<D>$ if $H$ rejects $<D,<D>>$ if $D$ rejects $<D>$
   Either way: Contradiction! $D$ cannot exist $\Rightarrow H$ cannot exist
Therefore, $A_{TM}$ is not a decidable language.

Undecidability Proof uses Diagonalization

Input strings

<table>
<thead>
<tr>
<th>List of TMs</th>
<th>Input strings</th>
<th>If $H$ exists</th>
<th>D outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>$&lt;M_1&gt;$</td>
<td>$M_1$</td>
<td>$&lt;M_1&gt;$</td>
</tr>
<tr>
<td>$M_2$</td>
<td>$&lt;M_2&gt;$</td>
<td>$M_2$</td>
<td>$&lt;M_2&gt;$</td>
</tr>
<tr>
<td>$M_3$</td>
<td>$&lt;M_3&gt;$</td>
<td>$M_3$</td>
<td>$&lt;M_3&gt;$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

D on $<M_i>$ accepts if and only if $M_i$ on $<M_i>$ rejects.
So, D on $<D>$ will accept if and only if D on $<D>$ rejects!
A contradiction $\Rightarrow H$ cannot exist!
Therefore, $A_{TM}$ is not a decidable language.
One Last Concept: Reducibility

- How do we show a new problem B is undecidable?
  - Idea: Show that $A_{TM}$ is reducible to the new problem B
    - What does this mean and how do we show this?
  - Show that if B was decidable, then you can use the decider for B as a subroutine to decide $A_{TM}$
    - Contradiction, therefore B must also be undecidable

The Halting Problem is Undecidable (Turing, 1936)

- Example: Halting Problem: Does TM M halt on input w?
  - Equivalent language: $A_H = \{ <M, w> \mid TM \text{ M halts on input } w \}$
  - Need to show $A_H$ is undecidable
  - We know $A_{TM} = \{ <M, w> \mid TM \text{ M accepts w} \}$ is undecidable
- Show $A_{TM}$ is reducible to $A_H$ (Theorem 5.1 in text)
  - Suppose $A_H$ is decidable ⇒ there’s a decider $M_H$ for $A_H$
  - Then, we can construct a decider $D_{TM}$ for $A_{TM}$:
    - On input $<M, w>$, run $M_H$ on $<M, w>$.
      - If $M_H$ rejects, then REJ (this takes care of M looping on w)
      - If $M_H$ accepts, then simulate M on w until M halts
      - If M accepts, then ACC input $<M, w>$; else REJ
    - $L(D_{TM}) = A_{TM} ⇒ A_{TM}$ is decidable! Contradiction ⇒ $A_H$ is undecidable
- E.g. 2: Show $E_{TM} = \{ <M> \mid M \text{ is a TM and } L(M) = \emptyset \}$ is undecidable (see Theorem 5.2 in the text)