Today's learning goals

- Trace high-level descriptions of algorithms for computational problems.
- Use counting arguments to prove the existence of unrecognizable (undecidable) languages.
- Use diagonalization in a proof of undecidability.
**Encoding input for TMs**

- By definition, TM inputs are **strings**

- To define TM M:

  "On input w ...
  1. ..
  2. ..
  3. ...

For inputs that aren't strings, we have to **encode the object** (represent it as a string) first

**Notation:**

- \(<O>\) is the **string** that represents (encodes) the object \(O\)
- \(<O_1, ..., O_n>\) is the **single** string that represents the tuple of objects \(O_1, ..., O_n\)
Encoding inputs

Payoff: problems we care about can be reframed as languages of strings

e.g. "Recognize whether a string is a palindrome."
\[ \{ w \mid w \in \{0,1\}^* \text{ and } w = w^R \} \]
e.g. "Check whether a string is accepted by a DFA."
\[ \{ \langle B, w \rangle \mid B \text{ is a DFA over } \Sigma, w \in \Sigma^*, \text{ and } w \text{ is in } L(B) \} \]
e.g. "Check whether the language of a PDA is infinite."
\[ \{ \langle A \rangle \mid A \text{ is a PDA and } L(A) \text{ is infinite} \} \]
Encoding inputs

**Payoff**: problems we care about can be reframed as languages of strings

e.g. "Recognize whether a string is a palindrome."

\[ \{ w \mid w \in \{0,1\}^* \text{ and } w = w^R \} \]

A. This set is regular and decidable.
B. This set is regular and not decidable
C. This set is nonregular and decidable
D. This set is nonregular and not decidable.
E. None of the above
Computational problems

A computational problem is **decidable** iff the language encoding the problem instances is decidable.
Computational problems

Sample computational problems and their encodings:

• \texttt{A}_{\text{DFA}} "Check whether a string is accepted by a DFA."
  \{ <B,w> | B is a DFA over \( \Sigma \), \( w \in \Sigma^* \), and \( w \) is in \( L(B) \) \}

• \texttt{E}_{\text{DFA}} "Check whether the language of a DFA is empty."
  \{ <A> | A is a DFA over \( \Sigma \), \( L(A) \) is empty \}

• \texttt{EQ}_{\text{DFA}} "Check whether the languages of two DFAs are equal."
  \{ <A, B> | A and B are DFA over \( \Sigma \), \( L(A) = L(B) \) \}

FACT: all of these problems are decidable!
Computational problems

Sample computational problems and their encodings:

• $A_{\text{PDA}}$: "Check whether a string is accepted by a PDA."
  \[ \{ <B,w> \mid B \text{ is a PDA over } \Sigma, \ w \in \Sigma^*, \text{ and } w \text{ is in } L(B) \} \]

• $E_{\text{PDA}}$: "Check whether the language of a PDA is empty."
  \[ \{ <A> \mid A \text{ is a PDA over } \Sigma, \ L(A) \text{ is empty} \} \]

• $E_{\text{QPDA}}$: "Check whether the languages of two PDAs are equal."
  \[ \{ <A, B> \mid A \text{ and } B \text{ are PDA over } \Sigma, \ L(A) = L(B) \} \]

FACT: some of these problems are decidable, and some are not!
Computational problems

Sample computational problems and their encodings:

- \( A_{TM} \) "Check whether a string is accepted by a TM."
  \[
  \{ <B, w> \mid B \text{ is a TM over } \Sigma, \ w \text{ in } \Sigma^*, \text{ and } w \text{ is in } L(B) \}
  \]

- \( E_{TM} \) "Check whether the language of a TM is empty."
  \[
  \{ <A> \mid A \text{ is a TM over } \Sigma, \ L(A) \text{ is empty} \}
  \]

- \( EQ_{TM} \) "Check whether the languages of two TMs are equal."
  \[
  \{ <A, B> \mid A \text{ and } B \text{ are TM over } \Sigma, \ L(A) = L(B) \}
  \]

FACT: all of these problems are undecidable!
Undecidable?

- There are many ways to prove that a problem *is* decidable.
- How do we find (and prove) that a problem *is not* decidable?
Before we proved the Pumping Lemma ...

We proved there was a set that was not regular because all regular sets are countable, while not all sets of strings are.
Why is the set of Turing-recognizable languages **countable**?

A. It's equal to the set of all TMs, which we showed is countable.
B. It's a subset of the set of all TMs, which we showed is countable.
C. Each Turing-recognizable language is associated with a TM, so there can be no more Turing-recognizable languages than TMs.
D. More than one of the above.
E. I don't know.
Satisfied?

• Maybe not …

• What's a specific example of a language that is not Turing-recognizable? or not Turing-decidable?

• Idea: consider a set that, were it to be Turing-decidable, would have to "talk" about itself, and contradict itself!
$A_{TM}$

Recall $A_{DFA} = \{<B,w> \mid B \text{ is a DFA and } w \text{ is in } L(B)\}$

Decider for this set simulates arbitrary DFA

$A_{TM} = \{<M,w> \mid M \text{ is a TM and } w \text{ is in } L(M)\}$

Decider for this set simulates arbitrary TMs
\( \text{A}_{\text{TM}} \)

\( \text{A}_{\text{TM}} = \{<M,w> \mid M \text{ is a TM and } w \text{ is in } L(M) \} \)

Define the TM \( N \) = "On input \( <M,w> \):

1. Simulate \( M \) on \( w \).
2. If \( M \) accepts, accept. If \( M \) rejects, reject."
**A_{TM}**

A_{TM} = \{<M,w> | M is a TM and w is in L(M) \}

Define the TM N = "On input <M,w>:
1. Simulate M on w.
2. If M accepts, accept. If M rejects, reject."

What is L(N)? L(N) = A_{TM}  
N recognizes for A_{TM}  
N not a decider
Define the TM $N = \text{"On input } <M,w>:\text{ 
1. Simulate } M \text{ on } w. 
2. If } M \text{ accepts, accept. If } M \text{ rejects, reject."} 

Does $N$ decide $A_{TM}$? \text{ No}
Define the TM $N$ = "On input $<M,w>$:
1. Simulate $M$ on $w$.
2. If $M$ accepts, accept. If $M$ rejects, reject."

**Conclude:** $A_{TM}$ is Turing-recognizable.

Is it decidable?
Assume, towards a contradiction, that it is.

Call $M_{ATM}$ the decider for $A_{TM}$:

For every TM $M$ and every string $w$,

- Computation of $M_{ATM}$ on $<M, w>$ halts and accepts if $w$ is in $L(M)$.
- Computation of $M_{ATM}$ on $<M, w>$ halts and rejects if $w$ is not in $L(M)$. 
Diagonalization proof: $A_{TM}$ not decidable  

Assume, towards a contradiction, that $M_{ATM}$ decides $A_{TM}$

Define the TM $D =$ "On input $<M>$:

1. Run $M_{ATM}$ on $<M, <M>>$. 
2. If $M_{ATM}$ accepts, reject; if $M_{ATM}$ rejects, accept."
Diagonalization proof: $A_{TM}$ not decidable  

Assume, towards a contradiction, that $M_{ATM}$ decides $A_{TM}$

Define the TM $D = \text{"On input } <M>:\$

1. Run $M_{ATM}$ on $<M, <M>>$. 
2. If $M_{ATM}$ accepts, reject; if $M_{ATM}$ rejects, accept."

Which of the following computations halt?
A. Computation of $D$ on $<X>$
B. Computation of $D$ on $<Y>$ where $Y$ is TM with $L(Y) = \Sigma^*$
C. Computation of $D$ on $<D>$
D. All of the above.
Diagonalization proof: $A_{\text{TM}}$ not decidable \textit{Sipser 4.11}

Assume, towards a contradiction, that $M_{\text{ATM}}$ decides $A_{\text{TM}}$

Define the TM $D = \text{"On input }<M>:\text{"}
1. Run $M_{\text{ATM}}$ on $<M, <M>>$.
2. If $M_{\text{ATM}}$ accepts, reject; if $M_{\text{ATM}}$ rejects, accept.

Consider running $D$ on input $<D>$. Because $D$ is a decider:
- either computation halts and accepts ...
- or computation halts and rejects ...
all TMs: \( M_1, N_0, M_3, \ldots \)
inputs: \( w_1, w_2, w_3, \ldots \)

previous proof is this proof when \( w_i : \langle M_i \rangle \)

\[
\begin{array}{c|ccc}
  & w_1 & w_2 & w_3 \\
\hline
M_1 & 1 & 0 & 1 \\
M_2 & 0 & 0 & 1 \\
M_3 & \vdots & \vdots & \vdots \\
\end{array}
\]

\[ A_{ij} = \text{cell row } M_i, \text{ column } w_j \]
\[ 1 \text{ if } w_j \in L(M_i) \]
\[ 0 \text{ if not} \]

claim: there is no TM \( M \)

such: \( M(w_j) \) accepts \( \iff A_{ij} = 0 \)

proof: \( M \neq M_i \) because

\( M(w_i) \neq M_i(w_i) \)

\( L : \{ w_i : M_i(w_i) \text{ doesn't accept} \} \)
Diagonalization proof: $A_{TM}$ not decidable (Sipser 4.11)

Assume, towards a contradiction, that $M_{ATM}$ decides $A_{TM}$.

Define the TM $D = \text{"On input } <M>:\n
1. \text{Run } M_{ATM} \\text{on } <M, <M>>.\n2. \text{If } M_{ATM} \text{accepts, reject; if } M_{ATM} \text{rejects, accept.}\n
Consider running $D$ on input $<D>$. Because $D$ is a decider:

- either computation halts and accepts ...
- or computation halts and rejects ...

Self-reference

"Is $<D>$ an element of $L(D)$?"
A TM

• Recognizable
• Not decidable

Fact: A language is decidable iff it and its complement are both recognizable.

Corollary: The complement of A TM is unrecognizable.

Question: does $\text{Fin}_2$ halts for all inputs?

Answer: don't know
Decidable vs. undecidable

Which of the following languages is **undecidable**?

A. $\text{INFINITE}_{\text{DFA}} = \{ <A> | A \text{ is a DFA and } L(A) \text{ is an infinite language} \}$

B. $\text{S}_{\text{TM}} = \{ <M> | M \text{ is a TM and } M \text{ has exactly 7 states} \}$

C. $\text{Rev}_{\text{DFA}} = \{ <B> | B \text{ is a DFA and for all strings } w, B \text{ accepts } w \text{ iff it accepts } w^R \}$

D. $\text{Rec}_{\text{TM}} = \{ <X> | X \text{ is a TM and } L(X) \text{ is recognizable} \}$

E. $\text{Dec}_{\text{TM}} = \{ <Y> | Y \text{ is a TM and } L(Y) \text{ is decidable} \}$
## So far

<table>
<thead>
<tr>
<th>Decidable</th>
<th>Recognizable (and not decidable)</th>
<th>Co-recognizable (and not decidable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{DFA}$</td>
<td>$A_{TM}$</td>
<td>$A_{TM}^C$</td>
</tr>
<tr>
<td>$E_{DFA}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$EQ_{DFA}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$INFINITE_{DFA}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Rev_{DFA}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$S_{TM}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Rec_{TM}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Do we have to diagonalize?

- Next time: comparing difficulty of problems.