Today's learning goals

- Define reductions from one problem to another.
- Use reductions to prove undecidability.
- Describe the difference between diagonalization and reduction.
- Relate recognizability, co-recognizability, and decidability.
- Prove that a set is not recognizable.

Sipser Ch 5.1

Review sessions: Wednesday 5/25, Thursday 6/2
Reduction?

A problem $P_1$ **reduces to** a problem $P_2$ if any solution for $P_2$ can be used to solve $P_1$.

**Strategy**: to prove that a problem is undecidable, prove that a problem we know to be undecidable reduces to it.
So far

<table>
<thead>
<tr>
<th>Decidable</th>
<th>Undecidable</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{DFA}$</td>
<td>$A_{TM}$</td>
</tr>
<tr>
<td>$E_{DFA}$</td>
<td>$\text{HALT}_{TM}$</td>
</tr>
<tr>
<td>$\text{EQ}_{DFA}$</td>
<td>$E_{TM}$</td>
</tr>
<tr>
<td>$\text{ALL}_{DFA}$ (HW)</td>
<td>$\text{REGULAR}_{TM}$</td>
</tr>
<tr>
<td></td>
<td>${ &lt;M&gt;</td>
</tr>
</tbody>
</table>

Give algorithm!

Diagonalization OR reduction
General approach

To prove that \( \{ <M> \mid M \text{ is a TM and } L(M) \text{ has property } P \} \) is undecidable

- Assume towards a contradiction that \( R \) is a decider for \( \{ <M> \mid M \text{ is a TM and } L(M) \text{ has } P \} \).
- Build decider for \( A_{TM} \) by: "On input \( <M,w> \)
  1. Construct a new TM \( X \) such that \( X \) has \( P \) iff \( w \) in \( L(M) \)
  2. Run \( R \) on \( <X> \): if accepts, accept; if rejects, reject."

Note: sometimes easier to build \( X \) so that \( X \) has \( P \) iff \( w \) not in \( L(M) \)
A new proof that this language is undecidable…

Can we show that $A_{TM}$ reduces to it?

Let $R$ be a decider for \{ $<M>$ $|$ M is a decider \}. Build a decider for $A_{TM}$ by: "On input $<M, w>$

1.
2.
3.
   "
A new proof that this language is undecidable...

Can we show that $A_{TM}$ reduces to it?

Let $R$ be a decider for \{ $<M> \mid M$ is a decider \}. Build a decider for $A_{TM}$ by: "On input $<M,w>$

1. Build machine $X$ so that $X$ is a decider iff $w$ is in $L(M)$.
2. Run $R$ on $<X>$.
3. If accepts, accept. If rejects, reject."
{ <M> | M is a decider }

A new proof that this language is undecidable...

Can we show that $A_{TM}$ reduces to it?

Let $R$ be a decider for \{ <M> | M is a decider \}. Build a decider for $A_{TM}$ by: "On input <M,w>
1. Build machine $X$ so that $X$ is a decider iff $w$ is in $L(M)$.
2. Run $R$ on <X>.
3. If accepts, accept. If rejects, reject."
Another example

\[ L = \{ <M> \mid M \text{ is a TM and } M \text{ accepts } w \text{ iff } M \text{ accepts } w^R \} \]

**Claim:** This set is undecidable.

What algorithm should we build?

A. An algorithm to decide \( A_{TM} \) using a subroutine that decides \( L \)
B. An algorithm that decides \( \overline{A_{TM}} \) using a subroutine that decides \( A_{TM} \)?
C. Both
D. Neither
E. I don't know.
Another example

\[ L = \{ <M> \mid M \text{ is a TM and } M \text{ accepts } w \text{ iff } M \text{ accepts } w^R \} \]

Claim: This set is undecidable.

Proof: Assume (towards a contradiction) that there is a TM \( R \) that decides \( L \). Define an algorithm: "On input \(<M,w>:\"

1. Build \( X \) such that \( L(X) \) is in \( L \) iff \( w \) is in \( L(M) \).
2. Run \( R \) on \( X \) …
3. …"
Another example

\[ L = \{ \langle M \rangle \mid M \text{ is a TM and } M \text{ accepts } w \text{ iff } M \text{ accepts } w^R \} \]

**Claim:** This set is undecidable.

**Proof:** Assume (towards a contradiction) that there is a TM \( R \) that decides \( L \). Define an algorithm:

1. Build \( X \) such that \( L(X) \) is in \( R \) iff \( w \) is in \( L(M) \).
2. Run \( R \) on \( X \) …
3. …"

Which of these is in \( L \)?

A. \( \langle M_1 \rangle \) where \( L(M_1) = \{ 0^n1 \mid n > 0 \} \)
B. \( \langle M_2 \rangle \) where \( L(M_2) = \{ 01 \} \)
C. \( \langle M_3 \rangle \) where \( L(M_3) = \{ 01, 10 \} \)
D. None of the above / more than one.
E. I don't know.
Another example

\[ L = \{ <M> \mid M \text{ is a TM and } M \text{ accepts } w \text{ iff } M \text{ accepts } w^R \} \]

**Claim:** This set is undecidable.

**Proof:**

**Goal** using \( M, w \) as parameters, build \( X \) so that \( L(X) = \{01, 10\} \) if \( w \) is in \( L(M) \) and \( L(X) = \{01\} \) if not.
Another example

$L = \{ <M> \mid M \text{ is a TM and } M \text{ accepts } w \text{ iff } M \text{ accepts } w^R \}$

Claim: This set is undecidable.

Proof:

Goal using $M,w$ as parameters, build $X$ so that $L(X) = \{01,10\}$ if $w$ is in $L(M)$ and $L(X) = \{01\}$ if not.

What should $X$ do on input $x$?

A. Ignore it, and run $M$ on $w$.
B. Compare $x$ to $w$.
C. Compare $x$ to $01$.
D. None of the above / more than one.
E. I don't know.
Another example

\[ L = \{ <M> \mid M \text{ is a TM and } M \text{ accepts } w \text{ iff } M \text{ accepts } w^R \} \]

**Goal** using \( M, w \) as parameters, build \( X \) so that \( L(X) = \{01, 10\} \) if \( w \) is in \( L(M) \) and \( L(X) = \{01\} \) if not. Let \( X = " \text{ On input } x, \)

1. If \( x \) is not 01 or 10, then reject.
2. If \( x = 01 \), accept.
3. If \( x = 10 \), run \( M \) on \( w \). If accept, accept; if reject, reject."
Another example

\[ L = \{ <M> \mid M \text{ is a TM and } M \text{ accepts } w \text{ iff } M \text{ accepts } w^R \} \]

**Goal** using \( M, w \) as parameters, build \( X \) so that \( L(X) = \{01, 10\} \) if \( w \) is in \( L(M) \) and \( L(X) = \{01\} \) if not. Let \( X = "\) On input \( x, \)
1. If \( x \) is not 01 or 10, then reject.
2. If \( x = 01 \), accept.
3. If \( x = 10 \), run \( M \) on \( w \). If accept, accept; if reject, reject."

If \( w \) is in \( L(M) \), \( L(X) = ? \)
Another example

\[ L = \{ <M> | \text{M is a TM and M accepts } w \text{ iff } M \text{ accepts } w^R \} \]

**Goal** using \( M, w \) as parameters, build \( X \) so that \( L(X) = \{01, 10\} \) if \( w \) is in \( L(M) \) and \( L(X) = \{01\} \) if not. Let \( X = \) "On input \( x \),

1. If \( x \) is not 01 or 10, then reject.
2. If \( x = 01 \), accept.
3. If \( x = 10 \), run \( M \) on \( w \). If accept, accept; if reject, reject."

If \( w \) is not in \( L(M) \), \( L(X) = ? \)
Reducing other problems?

\[ \text{INF}_{\text{TM}} = \{ <M> \mid M \text{ is TM and } L(M) \text{ is infinite} \} \]

To prove \( \text{INF}_{\text{TM}} \) is undecidable, we will reduce problems known to be undecidable to it. E.g.

- \( A_{\text{TM}} \): Input \(<M,w>\); Need to decide if \( w \) is in \( L(M) \).
- \( \text{HALT}_{\text{TM}} \): Input \(<M,w>\); Need to decide if \( M \) halts on \( w \).
- \( E_{\text{TM}} \): Input \(<M>\); need to decide if \( L(M) \) is empty.
Reducing other problems?

Let \( R \) decide \( \text{INF}_{\text{TM}} = \{ <M> | M \text{ is TM and } L(M) \text{ is infinite} \} \)

\( A_{\text{TM}} \): Input \( <M,w> \); Need to decide if \( w \) is in \( L(M) \).

\( M_{\text{ATM}} \): "On input \( <M,w> \)

1. Build \( X \) such that \( L(X) \) is infinite iff \( w \) is in \( L(M) \).
2. Run \( R \) on \( X \). If accepts, accept; if rejects, reject."

\( X \): "On input \( x \)

1. Run \( M \) on \( w \). If accepts, accept; if rejects, reject."
Reducing other problems?

Let \( R \) decide \( \text{INF}_{\text{TM}} = \{ <M> \mid M \text{ is TM and } L(M) \text{ is infinite} \} \)

\( A_{\text{TM}} \): Input \( <M,w> \); Need to decide if \( w \) is in \( L(M) \).

\( M_{\text{HALT}} \): "On input \( <M,w> \)

1. Build \( X \) such that \( L(X) \) is infinite iff _________.
2. Run \( R \) on \( X \). If accepts, accept; if rejects, reject."

\( X \): "On input \( x \)

1. Run \( M \) on \( w \). If _________."
Reducing other problems?

Let \( R \) decide \( \text{INF}_{\text{TM}} = \{ <M> | M \text{ is TM and } L(M) \text{ is infinite} \} \)

\( A_{\text{TM}} \): Input \( <M,w> \); Need to decide if \( w \) is in \( L(M) \).

\( M_{\text{ETM}} \): "On input \( <M> \)
1. Build \( X \) such that \( L(X) \) is infinite iff \( L(M) \) is not empty.
2. Run \( R \) on \( X \). If accepts, \textbf{reject}; if rejects, \textbf{accept}."

\( X \): "On input \( x \)
1. Run \( M \) on ____________________."
Last example

\[ \text{EQ}_{\text{TM}} = \{<M_1, M_2> \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2)\} \]

Claim: \( \text{EQ}_{\text{TM}} \) is undecidable.

How do we pick which problem to reduce?

Option 1: "stick with what we know" … \( A_{\text{TM}} \)

Try as an exercise … what machine \( X \) will you build?
Option 2: "The road less travelled" … \( E_{\text{TM}} \)

Given \( M_{\text{EQ}} \) deciding \( \text{EQ}_{\text{TM}} \), build \( M_{\text{ETM}} \): "On input \( <M> \),

1. Build machine \( M_{\text{rej}} \) that rejects all inputs.
2. Run \( M_{\text{EQ}} \) on \( <M_{\text{rej}}, M> \). If accepts, accept; if rejects, reject."
Unrecognizable sets

Decidable sets

Recognizable sets

$\text{HALT}^\text{TM}$

$A^\text{TM}$
Invariant

**Theorem:** A language is decidable iff it is Turing-recognizable and its complement is Turing-recognizable
Invariant

Theorem: A language is decidable iff it is Turing-recognizable and its complement is Turing-recognizable.

As a consequence of this theorem: if a language is undecidable and Turing-recognizable then,

A. we get a contradiction.
B. its complement must be decidable.
C. its complement must be recognizable.
D. its complement must be unrecognizable.
E. I don't know.
**Invariant**

**Theorem:** A language is decidable iff it is Turing-recognizable and its complement is Turing-recognizable.

**Strategy:** To prove that a language is unrecognizable
- Prove that it is undecidable
- Prove that its complement is recognizable.

**First example:** \( \overline{A_{TM}} \) is unrecognizable.
Proof of theorem

• Assume L is decidable.
• WTS that L is recognizable and co-recognizable.
Proof of theorem

• Assume $L$ is recognizable and co-recognizable.
• WTS that $L$ is decidable.
Consequence

Claim: Exactly one of $E_{TM}$ and its complement is recognizable.

Proof:

Why not both?
Which is?