Today we learn

- Decidability results
- Halting problem
- Emptiness for TM is undecidable

Section 4.2, 5.1
Decidability and Recognizability

Understanding the limits of decision problems

**Implementation:** algorithm that answers a decision problem, that is algorithm says YES whenever decision problem says YES.

- **Decidability:** there is an implementation that terminates for all inputs
- **Undecidability:** any implementation will loop for some inputs
- **Unrecognizability:** no implementation is possible
Decidability and Recognizability

Understanding the limits of decision problems

**Implementation:** algorithm that answers a decision problem, that is algorithm says YES whenever decision problem says YES.

- **Decidability:** there is an implementation that terminates for all inputs
- **Undecidability:** any implementation will loop for some inputs
- **Unrecognizability:** no implementation is possible

Technically we are learning

- Proving the correctness of algorithms
- Proving the termination of algorithms
- Proving non-trivial results (combining multiple theorems)
Corollary 4.23

$\overline{A_{TM}}$ is unrecognizable
Corollary 4.23: $\overline{A_{TM}}$ is unrecognizable

**Lemma** co_a_tm_not_recognizable:

$\sim$ Recognizable ($\text{compl } A_{tm}$).

Done in class...
Corollary 4.18
Some languages are unrecognizable
Corollary 4.18 Some languages are unrecognizable

Proof.
Corollary 4.18 Some languages are unrecognizable

Proof. An example of an unrecognizable language is: $\overline{A_{TM}}$
If $L$ is decidable, then $\overline{L}$ is decidable
On pen-and-paper proofs

**Theorem 4.22**

A language is decidable iff it is Turing-recognizable and co-Turing-recognizable.

In other words, a language is decidable exactly when both it and its complement are Turing-recognizable.

**Proof** We have two directions to prove. First, if $A$ is decidable, we can easily see that both $A$ and its complement $\overline{A}$ are Turing-recognizable. Any decidable language is Turing-recognizable, and the complement of a decidable language also is decidable.
Proof of Theorem 4.22 Taken from the book.

First, if $A$ is decidable, we can easily see that both $A$ and its complement $\overline{A}$ are Turing-recognizable.

- $A$ is decidable, then $A$ is recognizable by definition.
- $A$ is decidable, then $\overline{A}$ is recognizable? Why?

Any decidable language is Turing-recognizable,

- Yes, by definition.

and the complement of a decidable language also is decidable.

- Why?
If $L$ is decidable, then $\overline{L}$ is decidable

1. Let $M$ decide $L$.
2. Create a Turing machine that negates the result of $M$.

```
Definition inv M w :=
  mlet b ← Call m w in halt_with (negb b).
```

3. Show that $\text{inv } M$ recognizes
   \[
   \text{Inv}(L) = \{ w \mid M \text{ rejects } w \} 
   \]
4. Show that the result of $\text{inv } M$ for any word $w$ is the
   negation of running $M$ with $m$, where negation of
   accept is reject, reject is accept, and loop is loop.
5. The goal is to show that $\text{inv } M$ recognizes $\overline{L}$ and is
   decidable.

What about loops? If $M$ 
loops on some word $w$, then $\text{inv } M$ would also loop. How is does $\text{inv } M$ recognize $\overline{L}$?
If $L$ is decidable, then $\overline{L}$ is decidable

1. Let $M$ decide $L$.
2. Create a Turing machine that negates the result of $M$.

Definition $\text{inv } M \ w :=$
\begin{verbatim}
mlet b = Call m w in halt_with (negb b).
\end{verbatim}

3. Show that $\text{inv } M$ recognizes
\[ \text{Inv}(L) = \{w \mid M \text{ rejects } w \} \]

4. Show that the result of $\text{inv } M$ for any word $w$ is the negation of running $M$ with $m$, where negation of accept is reject, reject is accept, and loop is loop.

5. The goal is to show that $\text{inv } M$ recognizes $\overline{L}$ and is decidable.

What about loops? If $M$ loops on some word $w$, then $\text{inv } M$ would also loop. How is does $\text{inv } M$ recognize $\overline{L}$?

Recall that $L$ is decidable, so $M$ will never loop.
If $L$ is decidable, then $\overline{L}$ is decidable

Continuation...

Part 1. Show that $\text{inv } \text{M}$ recognizes $\overline{L}$

We must show that: If $M$ decides $L$ and $\text{inv } \text{M}$ recognizes $\text{Inv}(L)$, then $\text{inv } \text{M}$ is decidable.

It is enough to show that if $M$ decides $L$, then $\text{Inv}(L) = \overline{L}$.

Show proof $\text{inv\_compl\_equiv}$.

Part 2. Show that $\text{inv } \text{M}$ is a decider

Show proof $\text{decides\_to\_compl}$.
Chapter 5: Undecidability
$\textit{HALT}_\text{TM}$: Termination of TM

Will this TM halt given this input?

(The Halting problem)
**HALT\textsubscript{TM} is undecidable**

Theorem 5.1: HALT\_TM loops for some input

Set-based encoding

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

Function-based encoding

```python
def HALT_TM(M, w):
    return M halts on w
```

Proof

**Proof idea:** Given Turing machine acc, show that acc decides \( A_{TM} \).

```python
def acc(M, w):
    if HALT_TM(M, w):
        return M(w)
    else:
        return False
```
Theorem 5.1: Proof overview

Apply Thm 4.11 to (H) "acc decides $A_{TM}$" and reach a contradiction. To prove H:

1. Show that acc recognizes $\text{Acc}_D$
2. Show that $\text{Acc}_D = A_{TM}$ (why do we need this step?)
3. Show that acc is decidable
$\text{HALT}_{TM}$ is undecidable

Part 1. Show that acc recognizes $\text{Acc}_D$

1. Show that if acc w accepts, then $p \in \text{Acc}_D$, ie, $D$ accepts $\langle M, p \rangle$ and $M$ accepts $w$. 

\begin{verbatim}
1 Definition acc p :=
2 let (M, w) := decode_machine_input p in
3 mlet b <- Call D p in
4 if b then Call M w else REJECT.
\end{verbatim}
HALT<sub>TM</sub> is undecidable

Part 1. Show that acc recognizes Acc<sub>D</sub>

1. Show that if acc w accepts, then p ∈ Acc<sub>D</sub>, ie, D accepts ⟨M, p⟩ and M accepts w.
   - Case analysis on Call D <M, w>
**HALT\textsubscript{TM} is undecidable**

Part 1. Show that acc recognizes $\textbf{Acc}_D$

1. Show that if acc $w$ accepts, then $p \in \textbf{Acc}_D$, ie, $D$ accepts $\langle M, p \rangle$ and $M$ accepts $w$.
   1. Case analysis on Call $D \langle M, w \rangle$
      1. $D$ accepts $\langle M, w \rangle$, then we get that $M$ accepts $w$
HALT_T_M is undecidable

Part 1. Show that acc recognizes Acc_D

1. Show that if acc w accepts, then p ∈ Acc_D, ie, D accepts ⟨M,p⟩ and M accepts w.
   ○ Case analysis on Call D <M,w>
     1. D accepts <M,w>, then we get that M accepts w
     2. D rejects <M,w>, then contradiction

2. Show that if w ∈ Acc_D, then acc w accepts.
**HALT**$_{TM}$ is undecidable

Part 1. Show that acc recognizes $\text{Acc}_D$

1. Show that if acc $w$ accepts, then $p \in \text{Acc}_D$, ie, $D$ accepts $\langle M, p \rangle$ and $M$ accepts $w$.
   - Case analysis on Call $D <M,w>$
     1. $D$ accepts $<M,w>$, then we get that $M$ accepts $w$
     2. $D$ rejects $<M,w>$, then contradiction

2. Show that if $w \in \text{Acc}_D$, then acc $w$ accepts.
   - Given $D$ accepts $\langle M, w \rangle$ and $M$ accepts $w$, show that acc $w$ accepts
**HALT\textsubscript{TM} is undecidable**

Part 1. Show that acc recognizes \( \text{Acc}_D \)

1. Show that if acc \( w \) accepts, then \( p \in \text{Acc}_D \), i.e., \( D \) accepts \( \langle M, p \rangle \) and \( M \) accepts \( w \).
   - Case analysis on \( \text{Call } D \langle M, w \rangle \)
     1. \( D \) accepts \( \langle M, w \rangle \), then we get that \( M \) accepts \( w \)
     2. \( D \) rejects \( \langle M, w \rangle \), then contradiction

2. Show that if \( w \in \text{Acc}_D \), then acc \( w \) accepts.
   - Given \( D \) accepts \( \langle M, w \rangle \) and \( M \) accepts \( w \), show that acc \( w \) accepts
   - Rewrite each in code, get accept
$\text{HALT}_{\text{TM}}$ is undecidable

Part 2. Show that $\text{Acc}_D = \text{AT}_{TM}$

1. Show that if $\langle M, w \rangle \in \text{Acc}_D$, then $\langle M, p \rangle \in \text{AT}_{TM}$
**HALT\textsubscript{TM} is undecidable**

Part 2. Show that $\mathsf{Acc}_D = \mathsf{A}_{TM}$

1. Show that if $\langle M, w \rangle \in \mathsf{Acc}_D$, then $\langle M, p \rangle \in \mathsf{A}_{TM}$
   - We have $M$ accepts $w$ from $\langle M, p \rangle \in \mathsf{Acc}_D$
$\text{HALT}_{\text{TM}}$ is undecidable

Part 2. Show that $\text{Acc}_D = A_{\text{TM}}$

1. Show that if $\langle M, w \rangle \in \text{Acc}_D$, then $\langle M, p \rangle \in A_{\text{TM}}$
   - We have $M$ accepts $w$ from $\langle M, p \rangle \in \text{Acc}_D$

2. Show that if (i) $\langle M, w \rangle \in A_{\text{TM}}$, then $\langle M, w \rangle \in \text{Acc}_D$, i.e.
$\text{HALT}_{TM}$ is undecidable

Part 2. Show that $\text{Acc}_D = A_{TM}$

1. Show that if $\langle M, w \rangle \in \text{Acc}_D$, then $\langle M, p \rangle \in A_{TM}$
   - We have $M$ accepts $w$ from $\langle M, p \rangle \in \text{Acc}_D$

2. Show that if (i) $\langle M, w \rangle \in A_{TM}$, then $\langle M, w \rangle \in \text{Acc}_D$, ie $M$ accepts $w$ and $D$ accepts $\langle M, w \rangle$
$\textit{HALT}_{TM}$ is undecidable

Part 2. Show that $\text{Acc}_D = A_{TM}$

1. Show that if $\langle M, w \rangle \in \text{Acc}_D$, then $\langle M, p \rangle \in A_{TM}$
   - We have $M$ accepts $w$ from $\langle M, p \rangle \in \text{Acc}_D$

2. Show that if (i) $\langle M, w \rangle \in A_{TM}$, then $\langle M, w \rangle \in \text{Acc}_D$, ie $M$ accepts $w$ and $D$ accepts $\langle M, w \rangle$
   - We have that $M$ accepts $w$ from (i)
HALT_{TM} is undecidable

Part 2. Show that $\text{Acc}_D = A_{TM}$

1. Show that if $\langle M, w \rangle \in \text{Acc}_D$, then $\langle M, p \rangle \in A_{TM}$
   - We have $M$ accepts $w$ from $\langle M, p \rangle \in \text{Acc}_D$

2. Show that if (i) $\langle M, w \rangle \in A_{TM}$, then $\langle M, w \rangle \in \text{Acc}_D$, ie $M$ accepts $w$ and $D$ accepts $\langle M, w \rangle$
   - We have that $M$ accepts $w$ from (i)
   - We have that $D$ accepts $\langle M, w \rangle$ since $M$ halts.
HALT_{TM} is undecidable

Part 3. Show that acc is decidable

Proof by contradiction. Assume acc loops with \( p = \langle M, w \rangle \) and reach a contradiction.
$\text{HALT}_{\text{TM}}$ is undecidable

Part 3. Show that acc is decidable

Proof by contradiction. Assume acc loops with $p = \langle M, w \rangle$ and reach a contradiction. If acc loops with $p$, then $D$ accepts $p$ and $M$ loops with $w$, or $D$ loops with $p$ $^\dagger$
\( \text{HALT}_{TM} \) is undecidable

Part 3. Show that acc is decidable

Proof by contradiction. Assume acc loops with \( p = \langle M, w \rangle \) and reach a contradiction.

If acc loops with \( p \), then \( D \) accepts \( p \) and \( M \) loops with \( w \), or \( D \) loops with \( p \)

- If \( D \) accepts \( p \), then \( M \) halts with \( w \), which contradicts with \( M \) loops with \( w \)
$\text{HALT}_{TM}$ is undecidable

Part 3. Show that acc is decidable

Proof by contradiction. Assume acc loops with $p = \langle M, w \rangle$ and reach a contradiction. If acc loops with $p$, then $D$ accepts $p$ and $M$ loops with $w$, or $D$ loops with $p$ †

- If $D$ accepts $p$, then $M$ halts with $w$, which contradicts with $M$ loops with $w$
- If $D$ loops with $p$, we reach a contradiction because $D$ is a decider

†: Why?
$E_{\text{TM}}$: Emptiness of TM

(Is the language of this TM empty?)
Theorem 5.2: $E_{TM}$ is undecidable

Set-based

$E_{TM} = \{ \langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \}$

Function-based

\[
\text{def } E_{TM}(M):
\]
\[
\quad \text{return } L(M) = \{\}
\]

Proof overview: show that $acc$ decides $A_{TM}$

\[
\text{def } build_{M1}(M, w):
\]
\[
\quad \text{def } M1(x):
\]
\[
\quad \quad \text{if } x = w:
\]
\[
\quad \quad \quad \text{return } M \text{ accepts } w
\]
\[
\quad \quad \text{else:}
\]
\[
\quad \quad \quad \text{return False}
\]
\[
\quad \text{return } M1
\]

\[
\text{def } acc(M, w):
\]
\[
\quad b = E_{TM}(build_{M1}(M, w))
\]
\[
\quad \text{return } \neg b
\]

- $w \in L(M1) \iff \langle M1 \rangle \notin E_{TM}$
- $w \in L(M1) \iff w \in L(M)$
Theorem 5.2: $E_{TM}$ is undecidable

Proof follows by contradiction.
Theorem 5.2: \( E_{TM} \) is undecidable

Proof follows by contradiction.

1. Show that \( E_{TM} \) decidable implies \( A_{TM} \) decidable.
Theorem 5.2: $E_{TM}$ is undecidable

Proof follows by contradiction.

1. Show that $E_{TM}$ decidable implies $A_{TM}$ decidable.
2. Reach contradiction by applying Thm 4.11 to (1)
Theorem 5.2: $E_{TM}$ is undecidable

Proof follows by contradiction.

1. Show that $E_{TM}$ decidable implies $A_{TM}$ decidable.
2. Reach contradiction by applying Thm 4.11 to (1)

Goal: $E_{TM}$ decidable implies $A_{TM}$ decidable
Theorem 5.2: $E_{TM}$ is undecidable

Proof follows by contradiction.

1. Show that $E_{TM}$ decidable implies $A_{TM}$ decidable.
2. Reach contradiction by applying Thm 4.11 to (1)

Goal: $E_{TM}$ decidable implies $A_{TM}$ decidable

Let $D$ decide $E_{TM}$.

1. Show that acc recognizes $A_{TM}$
Theorem 5.2: $E_{TM}$ is undecidable

Proof follows by contradiction.

1. Show that $E_{TM}$ decidable implies $A_{TM}$ decidable.
2. Reach contradiction by applying Thm 4.11 to (1)

Goal: $E_{TM}$ decidable implies $A_{TM}$ decidable

Let $D$ decide $E_{TM}$.

1. Show that acc recognizes $A_{TM}$
   1. Show that $A_{TM} = Acc_D$ where $Acc_D = \{ \langle M, w \rangle \mid L(M1_{M,w}) \neq \emptyset \}$
      (e_tm_a_tm_spec)
Theorem 5.2: $E_{TM}$ is undecidable

Proof follows by contradiction.

1. Show that $E_{TM}$ decidable implies $A_{TM}$ decidable.
2. Reach contradiction by applying Thm 4.11 to (1)

Goal: $E_{TM}$ decidable implies $A_{TM}$ decidable

Let $D$ decide $E_{TM}$.

1. Show that acc recognizes $A_{TM}$
   1. Show that $A_{TM} = \text{Acc}_D$ where $\text{Acc}_D = \{ \langle M, w \rangle \mid L(M_{1M,w}) \neq \emptyset \}$
      (e_tm_a_tm_spec)
   2. Show that acc recognizes $\text{Acc}_D$ (E_tm_A_tm_recognizes)
Theorem 5.2: \( E_{TM} \) is undecidable

Proof follows by contradiction.

1. Show that \( E_{TM} \) decidable implies \( A_{TM} \) decidable.
2. Reach contradiction by applying Thm 4.11 to (1)

Goal: \( E_{TM} \) decidable implies \( A_{TM} \) decidable

Let \( D \) decide \( E_{TM} \).

1. Show that acc recognizes \( A_{TM} \)
   1. Show that \( A_{TM} = Acc_D \) where \( Acc_D = \{ \langle M, w \rangle \mid L(M_1, w) \neq \emptyset \} \) (e_tm_a_tm_spec)
   2. Show that acc recognizes \( Acc_D \) (E_tm_A_tm_recognizes)
2. Show that acc is a decider (decider_E_tm_A_tm)
Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = \text{Acc}_D$ where $\text{Acc}_D = \{ \langle M, w \rangle \mid L(M_{1,M,w}) \neq \emptyset \}$

Theorem not_empty_to_accept

1. Show that: If $L(M_{1,M,w}) \neq \emptyset$, then $M$ accepts $w$. 


Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = Acc_D$ where $Acc_D = \{ \langle M, w \rangle \mid L(M1, w) \neq \emptyset \}$

Theorem not_empty_toAccept

1. Show that: If $L(M1, w) \neq \emptyset$, then $M$ accepts $w$.
   - Case analysis on running $M$ with input $w$: 

Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = Acc_D$ where $Acc_D = \{ \langle M, w \rangle \mid L(M1_{M,w}) \neq \emptyset \}$

Theorem not_empty_to_accept

1. Show that: If $L(M1_{M,w}) \neq \emptyset$, then $M$ accepts $w$.
   - Case analysis on running $M$ with input $w$:
     - Case (a) $M$ accepts $w$: use assumption to conclude
Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = Acc_{D}$ where $Acc_{D} = \{\langle M, w \rangle \mid L(M_{1}, w) \neq \emptyset \}$

Theorem not_empty_to_accept

1. Show that: If $L(M_{1}, w) \neq \emptyset$, then $M$ accepts $w$.
   - Case analysis on running $M$ with input $w$:
     - Case (a) $M$ accepts $w$: use assumption to conclude
     - Case (b) $M$ rejects $w$: we can conclude that $L(M_{1}, w) = \emptyset$ from (b)
Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = \text{Acc}_D$ where $\text{Acc}_D = \{\langle M, w \rangle \mid L(M_1, w) \neq \emptyset\}$

Theorem not_empty_to_accept

1. Show that: If $L(M_1, w) \neq \emptyset$, then $M$ accepts $w$.
   - Case analysis on running $M$ with input $w$:
     - Case (a) $M$ accepts $w$: use assumption to conclude
     - Case (b) $M$ rejects $w$: we can conclude that $L(M_1, w) = \emptyset$ from (b)
     - Case (c) $M$ loops with $w$: same as above
Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = Acc_D$ where $Acc_D = \{ \langle M, w \rangle \mid L(M_1, w) \neq \emptyset \}$

Theorem accept_to_not_empty

2. Show that: If $M$ accepts $w$, then $L(M_1, w) \neq \emptyset$. 
Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = Acc_D$ where $Acc_D = \{ \langle M, w \rangle \mid L(M_1, w) \neq \emptyset \}$

Theorem accept_to_not_empty

2. Show that: If $M$ accepts $w$, then $L(M_1, w) \neq \emptyset$.
   1. Proof follows by contradiction: assume $L(M_1, w) = \emptyset$. 


Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = Acc_D$ where $Acc_D = \{ \langle M, w \rangle \mid L(M1_M, w) \neq \emptyset \}$

Theorem accept_to_not_empty

2. Show that: If $M$ accepts $w$, then $L(M1_M, w) \neq \emptyset$.
   1. Proof follows by contradiction: assume $L(M1_M, w) = \emptyset$.
   2. We know that $M1_M, w$ does not accept $w$ from (2.1)
Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = Acc_D$ where $Acc_D = \{ \langle M, w \rangle \mid L(M, w) \neq \emptyset \}$

Theorem accept_to_not_empty

2. Show that: If $M$ accepts $w$, then $L(M, w) \neq \emptyset$.
   1. Proof follows by contradiction: assume $L(M, w) = \emptyset$.
   2. We know that $M_{1M,w}$ does not accept $w$ from (2.1)
   3. To contradict 2.2, we show that $M_{1M,w}$ accepts $w$
Theorem 5.2: $E_{TM}$ is undecidable

Part 1.1: Show that $A_{TM} = \text{Acc}_D$ where $\text{Acc}_D = \{\langle M, w \rangle \mid L(M_{1\ M}, w) \neq \emptyset \}$

Theorem accept_to_not_empty

2. Show that: If $M$ accepts $w$, then $L(M_{1\ M}, w) \neq \emptyset$.
   1. Proof follows by contradiction: assume $L(M_{1\ M}, w) = \emptyset$.
   2. We know that $M_{1\ M}, w$ does not accept $w$ from (2.1)
   3. To contradict 2.2, we show that $M_{1\ M}, w$ accepts $w$
      1. Since $x = w$ and (2.1), then $M_{1\ M}, w$ accepts $w$