Undecidability

CSCI 338
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Construct TM $M_1$ on input $\langle x \rangle$:
   1. accept.
2. Construct TM $M_2$ on input $\langle y \rangle$:
   1. Run $N$ on $\omega$ and accept if $N$ does.
3. Run $H$ on $\langle M_1, M_2 \rangle$.
4. If $H$ accepts, accept. If $H$ rejects, reject.

If $N$ accepts $\omega$, then $M_1$ and $M_2$ have the same language ($\Sigma^*$). If $N$ does not accept $\omega$, then they have different languages. Thus $S$ decides $A_{TM}$. (bad!)
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. 

To show $EQ_{TM}$ is undecidable, use it to decide $E_{TM}$. 
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

1. We have a way ($H$) to test if two TMs have the same language. How could we use that to test if a TM’s language is empty?

Plan:?
Claim: $EQ_{TM} = \{\langle M, N \rangle : M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. We have a way ($H$) to test if two TMs have the same language. How could we use that to test if a TM’s language is empty?

Plan: Make a TM with an empty language and use $H$ to compare it to input to $E_{TM}$. 
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{ on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
\[ EQ_{TM} \]

Claim: \( EQ_{TM} = \{ \langle M, N \rangle : M, N \text{ are TMs and } L(M) = L(N) \} \) is undecidable.

Proof: Suppose \( EQ_{TM} \) is decidable and let TM \( H \) be its decider.

Build a TM \( S \) that decides \( E_{TM} \):

\[ S = \text{on input } \langle P \rangle \]

1. Construct TM \( M_2 \) on input \( \langle x \rangle \):
   1. reject.
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.

\[ L(M_2) = \emptyset \]
Claim: $EQ_{TM} = \{\langle M, N \rangle : M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   
   1. reject.

2. ?
Claim: $EQ_{TM} = \{\langle M, N \rangle : M, N \text{ are TMs and } L(M) = L(N) \}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.

2. Run $H$ on $\langle P, M_2 \rangle$. 
**EQ\textsubscript{TM}**

Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
2. Run $H$ on $\langle P, M_2 \rangle$.
3. If $H$ accepts, ____
   If $H$ rejects, ____.
$EQ_{TM}$

Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
   2. Run $H$ on $\langle P, M_2 \rangle$.
   3. If $H$ accepts, accept. If $H$ rejects, reject.
$EQ_{TM}$

Claim: $EQ_{TM} = \{\langle M, N \rangle : M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S$ = on input $\langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
2. Run $H$ on $\langle P, M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If...?
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
   2. Run $H$ on $\langle P, M_2 \rangle$.
   3. If $H$ accepts, accept. If $H$ rejects, reject.

If $L(P) = \emptyset$, ...?
**Claim:** $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

**Proof:** Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
2. Run $H$ on $\langle P, M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $L(P) = \emptyset$, $M_2$ and $P$ will have the same language (since $L(M_2) = \emptyset$) and...?
**$EQ_{TM}$**

Claim: $EQ_{TM} = \{(M, N): M, N$ are TMs and $L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.

2. Run $H$ on $\langle P, M_2 \rangle$.

3. If $H$ accepts, accept. If $H$ rejects, reject.

If $L(P) = \emptyset$, $M_2$ and $P$ will have the same language (since $L(M_2) = \emptyset$) and $S$ will accept.
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N) \}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
2. Run $H$ on $\langle P, M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $L(P) = \emptyset$, $M_2$ and $P$ will have the same language (since $L(M_2) = \emptyset$) and $S$ will accept. If $L(P) \neq \emptyset$, ...?
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
2. Run $H$ on $\langle P, M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $L(P) = \emptyset$, $M_2$ and $P$ will have the same language (since $L(M_2) = \emptyset$) and $S$ will accept. If $L(P) \neq \emptyset$, $M_2$ and $P$ will not have the same language and...?
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.

2. Run $H$ on $\langle P, M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $L(P) = \emptyset$, $M_2$ and $P$ will have the same language (since $L(M_2) = \emptyset$) and $S$ will accept. If $L(P) \neq \emptyset$, $M_2$ and $P$ will not have the same language and $S$ will reject.
Claim: $EQ_{TM} = \{\langle M, N \rangle: M, N \text{ are TMs and } L(M) = L(N)\}$ is undecidable.

Proof: Suppose $EQ_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $E_{TM}$:

$S = \text{on input } \langle P \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. reject.
2. Run $H$ on $\langle P, M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $L(P) = \emptyset$, $M_2$ and $P$ will have the same language (since $L(M_2) = \emptyset$) and $S$ will accept. If $L(P) \neq \emptyset$, $M_2$ and $P$ will not have the same language and $S$ will reject. Therefore, $S$ is a decider for $E_{TM}$, which is a contradiction, so $EQ_{TM}$ is undecidable.
Claim: $REGULAR_{TM} = \{\langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof:
**REGULAR_{TM}**

Claim: \( \text{REGULAR}_{TM} = \{ \langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \} \) is undecidable.

Proof: Suppose \( \text{REGULAR}_{TM} \) is decidable and let TM \( H \) be its decider.
Claim: \( \text{REGULAR}_{TM} = \{\langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular}\} \) is undecidable.

Proof: Suppose \( \text{REGULAR}_{TM} \) is decidable and let TM \( H \) be its decider.

Build a TM \( S \) that decides \( A_{TM} \):

\[
S = \text{on input } \langle N, \omega \rangle
\]

1. To show \( \text{REGULAR}_{TM} \) is undecidable, use it to decide \( A_{TM} \).
Claim: $\text{REGULAR}_{TM} = \{ \langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof: Suppose $\text{REGULAR}_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Plan: Build a TM whose language is regular if $N$ accepts $\omega$ and not regular if $N$ does not accept $\omega$. 
Claim: $\text{REGULAR}_{TM} = \{\langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof: Suppose $\text{REGULAR}_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{ on input } \langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:

   $L(M_2)$ is regular ⇐ $N$ accepts $\omega$

Plan: Build a TM whose language is regular if $N$ accepts $\omega$ and not regular if $N$ does not accept $\omega$. 
**REGULAR_{TM}**

Claim: $\text{REGULAR}_{TM} = \{\langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof: Suppose $\text{REGULAR}_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{ ?? ? ? \}$, accept.
   2. If $x \notin \{ ?? ? ? \}$, run $N$ on $\omega$ and accept if $N$ does.

Plan: Build a TM whose language is regular if $N$ accepts $\omega$ and not regular if $N$ does not accept $\omega.$
\textbf{REGULAR}_{TM}

Claim: \( REGULAR_{TM} = \{ \langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \} \) is undecidable.

Proof: Suppose \( REGULAR_{TM} \) is decidable and let TM \( H \) be its decider.

Build a TM \( S \) that decides \( A_{TM} \):

\[
S = \text{on input } \langle N, \omega \rangle
\]

1. Construct TM \( M_2 \) on input \( \langle x \rangle \):
   1. If \( x \in \{0^n1^n : n \geq 0\} \), accept.
   2. If \( x \notin \{0^n1^n : n \geq 0\} \), run \( N \) on \( \omega \) and accept if \( N \) does.

\( L(M_2) \) is regular \( \iff \) \( N \) accepts \( \omega \)

Plan: Build a TM whose language is regular if \( N \) accepts \( \omega \) and not regular if \( N \) does not accept \( \omega \).
Claim: $REGULAR_{TM} = \{\langle M \rangle: M$ is a TM and $L(M)$ is regular$\}$ is undecidable.

Proof: Suppose $REGULAR_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n: n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n: n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.

Plan: Build a TM whose language is regular if $N$ accepts $\omega$ and not regular if $N$ does not accept $\omega$. 
Claim: $\text{REGULAR}_{TM} = \{ \langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof: Suppose $\text{REGULAR}_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n : n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n : n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.

$L(M_2) = 0^n1^n \text{ or } \Sigma^*$

Plan: Build a TM whose language is regular if $N$ accepts $\omega$ and not regular if $N$ does not accept $\omega.$
**REGULAR\_TM**

Claim: \( \text{REGULAR}\_TM = \{\langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \} \) is undecidable.

Proof: Suppose \( \text{REGULAR}\_TM \) is decidable and let TM \( H \) be its decider.

Build a TM \( S \) that decides \( A\_TM \):

\[S = \text{on input } \langle N, \omega \rangle\]

1. Construct TM \( M_2 \) on input \( \langle x \rangle \):
   
   1. If \( x \in \{0^n1^n : n \geq 0\} \), accept.
   
   2. If \( x \notin \{0^n1^n : n \geq 0\} \), run \( N \) on \( \omega \) and accept if \( N \) does.

2. ?
Claim: $\text{REGULAR}_TM = \{\langle M \rangle: M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof: Suppose $\text{REGULAR}_TM$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n: n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n: n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.

2. Run $H$ on $\langle M_2 \rangle$. 

$\text{REGULAR}_TM$
Claim: $REGULAR_{TM} = \{ \langle M \rangle : M$ is a TM and $L(M)$ is regular$\}$ is undecidable.

Proof: Suppose $REGULAR_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n : n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n : n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.

2. Run $H$ on $\langle M_2 \rangle$.

3. If $H$ accepts, accept. If $H$ rejects, reject.
**REGULAR_{TM}**

Claim: $\text{REGULAR}_{TM} = \{\langle M \rangle: M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof: Suppose $\text{REGULAR}_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S = \text{on input } \langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n: n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n: n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.
2. Run $H$ on $\langle M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $N$ accepts $\omega$, $L(M_2) = \Sigma^*$ (regular).
**REGULAR\textsubscript{TM}**

Claim: $\text{REGULAR}_{\text{TM}} = \{\langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular}\}$ is undecidable.

Proof: Suppose $\text{REGULAR}_{\text{TM}}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{\text{TM}}$:

$S =$ on input $\langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n : n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n : n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.
2. Run $H$ on $\langle M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $N$ accepts $\omega$, $L(M_2) = \Sigma^*$ (regular). If $N$ does not accept $\omega$, $L(M_2) = \{0^n1^n : n \geq 0\}$ (not regular).
**REGULAR$_{TM}$**

Claim: $REGULAR_{TM} = \{\langle M \rangle : M$ is a TM and $L(M)$ is regular$\}$ is undecidable.

Proof: Suppose $REGULAR_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$S =$ on input $\langle N, \omega \rangle$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n : n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n : n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.

2. Run $H$ on $\langle M_2 \rangle$.

3. If $H$ accepts, accept. If $H$ rejects, reject.

If $N$ accepts $\omega$, $L(M_2) = \Sigma^*$ (regular). If $N$ does not accept $\omega$, $L(M_2) = \{0^n1^n : n \geq 0\}$ (not regular). So, deciding if $L(M_2)$ is regular will determine if $N$ accepts $\omega$. 
Claim: $REGULAR_{TM} = \{ \langle M \rangle : M \text{ is a TM and } L(M) \text{ is regular} \}$ is undecidable.

Proof: Suppose $REGULAR_{TM}$ is decidable and let TM $H$ be its decider.

Build a TM $S$ that decides $A_{TM}$:

$$S = \text{on input } \langle N, \omega \rangle$$

1. Construct TM $M_2$ on input $\langle x \rangle$:
   1. If $x \in \{0^n1^n : n \geq 0\}$, accept.
   2. If $x \notin \{0^n1^n : n \geq 0\}$, run $N$ on $\omega$ and accept if $N$ does.
2. Run $H$ on $\langle M_2 \rangle$.
3. If $H$ accepts, accept. If $H$ rejects, reject.

If $N$ accepts $\omega$, $L(M_2) = \Sigma^*$ (regular). If $N$ does not accept $\omega$, $L(M_2) = \{0^n1^n : n \geq 0\}$ (not regular). So, deciding if $L(M_2)$ is regular will determine if $N$ accepts $\omega$. Therefore, $S$ is a decider for $A_{TM}$, so $REGULAR_{TM}$ is undecidable.
When in doubt use $A_{TM}$ !!!
Unrecognizable Language

Claim: A language is decidable $\iff$ it and its complement are Turing-recognizable.

Proof:
Claim: A language is decidable $\iff$ it and its complement are Turing-recognizable.

Proof: $\Rightarrow$ If a language is decidable, its complement is also decidable (just reverse accept/reject conditions) and decidable languages are recognizable.

Given decider $T$ for $A$, make decider for $\bar{A}$:

$M = \text{on input } \omega$

1. Run $T$ on $\omega$.
2. If $T$ accepts, reject. If $T$ rejects, accept.
Unrecognizable Language

Claim: A language is decidable ⇔ it and its complement are Turing-recognizable.

Proof: ⇒ If a language is decidable, its complement is also decidable (just reverse accept/reject conditions) and decidable languages are recognizable.

⇐ If $A$ and $\overline{A}$ are both Turing-recognizable, let $M_1$ and $M_2$ be recognizers for $A$ and $\overline{A}$. 
Unrecognizable Language

Claim: A language is decidable $\iff$ it and its complement are Turing-recognizable.

Proof: $\implies$ If a language is decidable, its complement is also decidable (just reverse accept/reject conditions) and decidable languages are recognizable.

$\impliedby$ If $A$ and $\overline{A}$ are both Turing-recognizable, let $M_1$ and $M_2$ be recognizers for $A$ and $\overline{A}$. Consider the following TM:

$$M = \text{on input } \omega$$

1. Run both $M_1$ and $M_2$ on $\omega$ in parallel (alternate instructions).
2. If $M_1$ accepts, accept. If $M_2$ accepts, reject.
Unrecognizable Language

Claim: A language is decidable $\iff$ it and its complement are Turing-recognizable.

Proof: $\implies$ If a language is decidable, its complement is also decidable (just reverse accept/reject conditions) and decidable languages are recognizable.

$\impliedby$ If $A$ and $\overline{A}$ are both Turing-recognizable, let $M_1$ and $M_2$ be recognizers for $A$ and $\overline{A}$. Consider the following TM:

$M =$ on input $\omega$
1. Run both $M_1$ and $M_2$ on $\omega$ in parallel (alternate instructions).
2. If $M_1$ accepts, accept. If $M_2$ accepts, reject.

Since $\omega \in A$ or $\overline{A}$, $M_1$ or $M_2$ must accept (halts on input). Thus, $M$ is a decider for $A$. 
Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof:
Unrecognizable Language

Claim: \( \overline{HALT}_{TM} = \{ \langle M, \omega \rangle : M \text{ is a TM and } M \text{ does not halt on } \omega \} \) is not Turing-recognizable.

Proof: Suppose \( \overline{HALT}_{TM} \) was Turing-recognizable. Let \( T \) be its recognizer (i.e., \( ???? \)).
Unrecognizable Language

Claim: \( \overline{HALT_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \} \) is not Turing-recognizable.

Proof: Suppose \( \overline{HALT_{TM}} \) was Turing-recognizable. Let \( T \) be its recognizer (i.e., \( T \) will accept if a TM does not halt on some input).
Unrecognizable Language

Claim: $\text{HALT}_{TM} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: Suppose $\text{HALT}_{TM}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle M, \omega \rangle$: 
Unrecognizable Language

Claim: $\text{HALT}_{TM} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: Suppose $\text{HALT}_{TM}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.
Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega \rangle: M$ is a TM and $M$ does not halt on $\omega \}$ is not Turing-recognizable.

Proof: Suppose $\overline{HALT_{TM}}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.

???
Unrecognizable Language

Claim: $\text{HALT}_{TM} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: Suppose $\text{HALT}_{TM}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.

$\text{HALT}_{TM}$ recognizer!
Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: Suppose $\overline{HALT_{TM}}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.

Consider $V$ on $\langle N, \omega \rangle$: $\overline{HALT_{TM}}$ recognizer!
Unrecognizable Language

Claim: $\overline{\text{HALT}_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: Suppose $\overline{\text{HALT}_{TM}}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.

Consider $V$ on $\langle N, \omega \rangle$:
1. Run $T$ on $\langle N, \omega \rangle$ and run $S$ on $\langle N, \omega \rangle$ in parallel.
Unrecognizable Language

Claim: $\overline{\text{HALT}_{TM}} = \{\langle M, \omega \rangle : M$ is a TM and $M$ does not halt on $\omega \}$ is not Turing-recognizable.

Proof: Suppose $\overline{\text{HALT}_{TM}}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.

Consider $V$ on $\langle N, \omega \rangle$:
1. Run $T$ on $\langle N, \omega \rangle$ and run $S$ on $\langle N, \omega \rangle$ in parallel.
2. If $T$ accepts, reject. If $S$ accepts, accept.
Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: Suppose $\overline{HALT_{TM}}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.

$\overline{HALT_{TM}}$ recognizer!

Consider $V$ on $\langle N, \omega \rangle$:
1. Run $T$ on $\langle N, \omega \rangle$ and run $S$ on $\langle N, \omega \rangle$ in parallel.
2. If $T$ accepts, reject. If $S$ accepts, accept.
Unrecognizable Language

Claim: $\text{HALT}_{TM} = \{\langle M, \omega \rangle : M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: Suppose $\text{HALT}_{TM}$ was Turing-recognizable. Let $T$ be its recognizer (i.e., $T$ will accept if a TM does not halt on some input).

Consider $S$ on $\langle N, \omega \rangle$:
1. Run $N$ on $\omega$.
2. accept.

Consider $V$ on $\langle N, \omega \rangle$:
1. Run $T$ on $\langle N, \omega \rangle$ and run $S$ on $\langle N, \omega \rangle$ in parallel.
2. If $T$ accepts, reject. If $S$ accepts, accept.
Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof:

A language is decidable $\iff$ it and its complement are Turing-recognizable.
Unrecognizable Language

Claim: $\overline{\text{HALT}_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: $\text{HALT}_{TM}$ is not decidable
Unrecognizable Language

Claim: \(\overline{HALT_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}\) is not Turing-recognizable.

Proof: \(HALT_{TM}\) is not decidable \(\Rightarrow\) \(HALT_{TM}\) and \(\overline{HALT_{TM}}\) cannot both be Turing-recognizable (otherwise \(HALT_{TM}\) would be decidable).

A language is decidable \(\iff\) it and its complement are Turing-recognizable.
Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega \rangle: M \text{ is a TM and } M \text{ does not halt on } \omega \}$ is not Turing-recognizable.

Proof: $HALT_{TM}$ is not decidable $\implies$ $HALT_{TM}$ and $\overline{HALT_{TM}}$ cannot both be Turing-recognizable (otherwise $HALT_{TM}$ would be decidable). Since $HALT_{TM}$ is Turing-recognizable, $\overline{HALT_{TM}}$ cannot be Turing-recognizable.

A language is decidable $\iff$ it and its complement are Turing-recognizable.
Computability Hierarchy

- Regular
- Context-Free
- Turing-recognizable
- decidable
- Turing-

\( \text{HALT}_{TM} \)
Beyond Decidability

What if $HALT_{TM}$ were “decidable”?
Beyond Decidability

What if $HALT_{TM}$ were “decidable”? 

Goldbach’s Conjecture:
• 280-year-old open problem.
• Every integer $\geq 2$ is sum of two primes.
Beyond Decidability

What if $HALT_{TM}$ were “decidable”?

Goldbach’s Conjecture:
- 280-year-old open problem.
- Every integer $\geq 2$ is sum of two primes.

Consider $G$ on $\langle x \rangle$:
1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to $n$, reject.
3. Increment $n$ and loop to step 1.
public boolean G() {
    int i = 2;
    while (true) {
        boolean found = false;
        for (int n = 1; n < i; n++) {
            for (int m = 1; m < i; m++) {
                if (isPrime(n) && isPrime(m) && m + n == i) {
                    found = true;
                }
            }
        }
        if (!found) {
            return false;
        }
        i++;
    }
}
Beyond Decidability

What if $HALT_{TM}$ were “decidable”?

Goldbach’s Conjecture:
- 280-year-old open problem.
- Every integer $\geq 2$ is sum of two primes.

Consider $G$ on $\langle x \rangle$:
1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to $n$, reject.
3. Increment $n$ and loop to step 1.

What does it mean if $G$ halts?
What does it mean if $G$ does not halt?
Beyond Decidability

What if $HALT_{TM}$ were “decidable”?

Goldbach’s Conjecture:
- 280-year-old open problem.
- Every integer $\geq 2$ is sum of two primes.

Consider $G$ on $\langle x \rangle$:
1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to $n$, reject.
3. Increment $n$ and loop to step 1.

What does it mean if $G$ halts?  **Goldbach’s conjecture is false!**
What does it mean if $G$ does not halt?  **Goldbach’s conjecture is true!**
Beyond Decidability

What if $HALT_{TM}$ were “decidable”?

Goldbach’s Conjecture:
- 280-year-old open problem.
- Every integer $\geq 2$ is sum of two primes.

Consider $G$ on $\langle x \rangle$:
1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to $n$, reject.
3. Increment $n$ and loop to step 1.

What does it mean if $G$ halts? Goldbach’s conjecture is false!
What does it mean if $G$ does not halt? Goldbach’s conjecture is true!

Turns out you can do this for lots of open problems over natural numbers (twin prime conjecture,...)