Reductions
The Limits of Computability

- Regular Languages
- DCFLs
- CFLs
- Decidable Languages
- Recognizable Languages

- All Languages
- $HALT$
- $L_D$
- $A_{TM}$
- $\overline{A}_{TM}$
Finding Unsolvable Problems

• Last time, we found five unsolvable problems.
• We directly proved that $L_D$ was unrecognizable, then used this fact to show four other languages were either undecidable or unrecognizable.
• In general, to prove that a problem is unsolvable (not R or not RE), we don't directly show that it is unsolvable.
• Instead, we show how a solution to that problem would let us solve an unsolvable problem.
Reductions

Problem A

Can be converted to

Problem B
Reductions

Problem A ➔ Can be converted to ➔ Problem B

Can be used to solve
If any instance of $A$ can be converted into an instance of $B$, we say that $A$ reduces to $B$. 

**Reductions**

Problem A

Can be converted to

Can be used to solve

Problem B
Reductions

If any instance of A can be converted into an instance of B, we say that A reduces to B.
If any instance of A can be converted into an instance of B, we say that A reduces to B.
Reductions

If any instance of A can be converted into an instance of B, we say that A reduces to B.

CNF-SAT  → Can be converted to  ←  3SAT
           Can be used to solve

reduces
Reductions

If any instance of A can be converted into an instance of B, we say that A reduces to B.
Reductions

If any instance of A can be converted into an instance of B, we say that A reduces to B.
Reductions

If any instance of $A$ can be converted into an instance of $B$, we say that $A$ reduces to $B$. 

Any RE Language Can be converted to $A_{TM}$ 

Can be used to solve
If any instance of A can be converted into an instance of B, we say that A \textit{reduces} to B.
Defining Reductions

- A reduction from \(A\) to \(B\) is a function \(f : \Sigma_1^* \rightarrow \Sigma_2^*\) such that

\[
\text{For any } w \in \Sigma_1^*, \ w \in A \iff f(w) \in B
\]
Defining Reductions

- A **reduction** from $A$ to $B$ is a function $f : \Sigma_1^* \rightarrow \Sigma_2^*$ such that

$$\text{For any } w \in \Sigma_1^*, \text{ } w \in A \iff f(w) \in B$$
Defining Reductions

- A reduction from A to B is a function $f : \Sigma_1^* \rightarrow \Sigma_2^*$ such that

For any $w \in \Sigma_1^*$, $w \in A$ iff $f(w) \in B$
Defining Reductions

- A **reduction** from $A$ to $B$ is a function $f : \Sigma_1^* \rightarrow \Sigma_2^*$ such that

  For any $w \in \Sigma_1^*$, $w \in A$ iff $f(w) \in B$

- Every $w \in A$ maps to some $f(w)$ in $B$.
- Every $w \notin A$ maps to some $f(w)$ not in $B$.
- $f$ does not have to be injective or surjective.
The Problem with this Definition

- Under this definition, *any* language $L_1$ reduces to *any* language $L_2$ unless $L_2 = \emptyset$ or $\Sigma^*$.

- Proof sketch: Since $L_2 \neq \emptyset$ and $L_2 \neq \Sigma^*$, there is some $w_{\text{yes}} \in L_2$ and some $w_{\text{no}} \notin L_2$.

- Define $f : \Sigma_1^* \rightarrow \Sigma_2^*$ as follows:
  - If $w \in L_1$, $f(w) = w_{\text{yes}}$.
  - If $w \notin L_2$, $f(w) = w_{\text{no}}$.

- Then $f$ is a reduction from $L_1$ to $L_2$. 
The Problem with this Definition

- This general reduction is mathematically well-defined, but might be impossible to actually compute!

- For example, let's reduce $L_D$ to $0^*1^*$.

- Take $w_{yes} = 01$, $w_{no} = 10$.

- Then $f(w)$ is defined as
  - If $w \in L_D$, $f(w) = 01$.
  - If $w \notin L_D$, $f(w) = 10$.

- There is no TM that can actually evaluate the function $f(w)$ on all inputs, since no TM can determine whether or not $w \in L_D$. 
Computable Functions

- To fix our definition, we need to introduce the idea of a computable function.

- A function \( f : \Sigma_1^* \rightarrow \Sigma_2^* \) is called a **computable function** if there is some TM \( M \) with the following behavior:

  "On input \( w \):
  
  Determine the value of \( f(w) \).
  
  Write \( f(w) \) on the tape.
  
  Move the tape head back to the far left.
  
  Halt."
Computable Functions

\[ f(w) = ww \]
Computable Functions

\[ f(w) = ww \]
Computable Functions

\[ f(w) = \begin{cases} 
2^{nm} & \text{if } w = 0^n1^m \\
\varepsilon & \text{otherwise}
\end{cases} \]
Computable Functions

\[ f(w) = \begin{cases} 2^{nm} & \text{if } w = 0^n1^m \\ \epsilon & \text{otherwise} \end{cases} \]
Mapping Reductions

- A function $f : \Sigma_1^* \to \Sigma_2^*$ is called a **mapping reduction** from A to B if
  - For any $w \in \Sigma_1^*$ we have that $w \in A$ iff $f(w) \in B$.
  - $f$ is a computable function.
- Intuitively, a mapping reduction from A to B says that a computer can transform any instance of A into an instance of B such that the answer to B is the answer to A.
Mapping Reducibility

- If there is a mapping reduction from \( A \) to \( B \), we say that \( A \) is **mapping reducible** to \( B \).
- Notation: \( A \leq_M B \) iff \( A \) is mapping reducible to \( B \).
- There is a very good reason for this notation, which we'll see in a moment.
- Two nice properties:
  - For any language \( A \), \( A \leq_M A \). *(why?)*
  - If \( A \leq_M B \) and \( B \leq_M C \), then \( A \leq_M C \). *(why?)*
Why Mapping Reducibility Matters

• **Theorem:** If $A$ is mapping reducible to $B$ ($A \leq^M B$) and $B$ is decidable, then $A$ is decidable.

• **Theorem:** If $A$ is mapping reducible to $B$ ($A \leq^M B$) and $B$ is recognizable, then $A$ is recognizable.

• Justification for the notation: $A \leq^M B$ informally means “$A$ is not harder than $B$.”
Why Mapping Reducibility Matters

- **Theorem**: If $A$ is undecidable and $A \leq_M B$, then $B$ is undecidable.
- **Theorem**: If $A$ is unrecognizable and $A \leq_M B$, then $B$ is unrecognizable.
- More justification for the notation: $A \leq_M B$ means “$B$ is at least as hard as $A$.”
Why Mapping Reducibility Matters

If this one is “easy” (R or RE)...

...then this one is “easy” (R or RE) too.
Why Mapping Reducibility Matters

If this one is “hard” (not R or not RE)...

\[ A \leq_{M} B \]

... then this one is “hard” (not R or not RE) too.
Sketch of the Proof

Machine H

$H = \text{"On input } w:\text{ Compute } f(w). \text{ Run } M \text{ on } f(w). \text{ If } M \text{ accepts } f(w), \text{ accept } w. \text{ If } M \text{ rejects } f(w), \text{ reject } w."$

$H \text{ accepts } w \iff M \text{ accepts } f(w) \iff f(w) \in B \iff w \in A$
Sketch of the Proof

- If $A \leq^m B$, then any decider/recognizer for $B$ can be converted into a decider/recognizer for $A$.

- Consequently, if $B$ is $R$ or $RE$, we can show that $A$ is $R$ or $RE$ as well.

- If $A$ is not $R$ or not $RE$, then $B$ cannot be $R$ or $RE$ because otherwise we could build a decider or recognizer for $A$. 
Using Reductions
Using Reductions

- Recall: The language $A_{TM}$ is defined as
  
  $$A_{TM} = \{ \langle M, w \rangle \mid w \in \mathcal{L}(M) \}$$

- Last time, we proved that $A_{TM} \in RE - R$ by showing that a decider for $A_{TM}$ could be converted into a decider for the diagonalization language $L_D$.

- Let's see an alternate proof that $A_{TM}$ is undecidable by using reductions.
The Complement of $A_{TM}$

- Recall: if $A_{TM} \in R$, then $\overline{A}_{TM} \in R$ as well.

- To show that $A_{TM}$ is undecidable, we will prove that the complement of $A_{TM}$ (denoted $\overline{A}_{TM}$) is unrecognizable.

- The language $\overline{A}_{TM}$ is the following:

\[
\overline{A}_{TM} = \{ x \mid x \neq \langle M, w \rangle \text{ for any TM } M \text{ and string } w \text{ or } x = \langle M, w \rangle \text{ and } w \notin \mathcal{L}(M) \}
\]
Recall: The diagonalization language $L_D$ is the language

\[ L_D = \{ \langle M \rangle \mid \langle M \rangle \notin \mathcal{L}(M) \} \]

We directly established that $L_D \not\in \text{RE}$ using a diagonal argument.

If we can show that $L_D \leq \overline{A}_{\text{TM}}$, then we have proven that $\overline{A}_{\text{TM}} \not\in \text{RE}$. 
Where We're Going

Machine $H$

$w$ \rightarrow Compute $f$ \rightarrow $f(w)$ \rightarrow Machine for $\overline{A}_{TM}$

Goal: Choose our function $f(w)$ such that this machine $H$ is a recognizer for $L_D$. 

Machine $R$

YES \rightarrow \ NO
\[ L_D \leq_M \overline{A}_{TM} \]

- Goal: Find a computable function \( f \) such that
\[
\langle M \rangle \in L_D \text{ iff } f(\langle M \rangle) \in \overline{A}_{TM}
\]
- Simplifying this using the definition of \( L_D \)
\[
\langle M \rangle \notin \mathcal{L}(M) \text{ iff } f(\langle M \rangle) \in \overline{A}_{TM}
\]
- Let's assume that \( f(\langle M \rangle) \) has the form \( \langle M', w \rangle \) for some TM \( M' \) and string \( w \). This means that
\[
\langle M \rangle \notin \mathcal{L}(M) \text{ iff } \langle M', w \rangle \in \overline{A}_{TM}
\]
\[
\langle M \rangle \notin \mathcal{L}(M) \text{ iff } w \notin \mathcal{L}(M')
\]
- If we can choose \( w \) and \( M' \) such that the above is true, we will have our reduction from \( L_D \) to \( \overline{A}_{TM} \).
- Choose \( M' = M, w = \langle M \rangle \).
What We Just Did

Compute \( f \) \( \langle M, \langle M \rangle \rangle \)

Run \( R \) on \( \langle M, \langle M \rangle \rangle \).
If \( R \) accepts \( \langle M, \langle M \rangle \rangle \), accept \( \langle M \rangle \).
If \( R \) rejects \( \langle M, \langle M \rangle \rangle \), reject \( \langle M \rangle \).

\( H \) accepts \( \langle M \rangle \) iff \( R \) accepts \( \langle M, \langle M \rangle \rangle \) iff \( \langle M, \langle M \rangle \rangle \in \overline{A_{TM}} \) iff \( \langle M \rangle \notin L(M) \) iff \( \langle M \rangle \in L_D \)

\( H = \text{"On input } \langle M \rangle: \)
- Compute \( \langle M, \langle M \rangle \rangle \).
- Run \( R \) on \( \langle M, \langle M \rangle \rangle \).
- If \( R \) accepts \( \langle M, \langle M \rangle \rangle \), accept \( \langle M \rangle \).
- If \( R \) rejects \( \langle M, \langle M \rangle \rangle \), reject \( \langle M \rangle \)."
$L_D \leq M \overline{\mathcal{A}}_{TM}$

- Let's define our function as follows:
  \[ f(\langle M \rangle) = \langle M, \langle M \rangle \rangle \]

- We're not done yet... what do we do if the input string doesn't have the form $\langle M \rangle$ for some TM $M$?

- In that case, we need to map that string to some TM/string pair that does accept.

- Let's pick some machine $S$ that accepts everything, then set $f(w) = \langle S, \epsilon \rangle$ in that case.
  - Could pick any string $w$, but $\epsilon$ is a nice one!
$L_D \leq_M \overline{A}_{TM}$

- The final version of our function $f$ is defined here:
  - $f(w) = \langle M, \langle M \rangle \rangle$, if $w = \langle M \rangle$ for some TM $M$.
  - $f(w) = \langle S, \varepsilon \rangle$ otherwise, where $S$ always accepts.
- It's reasonable to assume that $f$ is computable; details are left as an exercise.
- If we can formally prove that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then we have that $L_D \leq_M \overline{A}_{TM}$. Thus $\overline{A}_{TM} \notin RE$. 
Theorem: $L_D \leq_M \overline{A_{TM}}$. 

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $A_{TM}$. Consider the function $f$ defined as follows:

$f(w) = \langle M, \langle M \rangle \rangle$ if $w$ is an encoding of a TM $M$.

$f(w) = \langle S, \epsilon \rangle$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in A_{TM}$, then $f$ is a mapping reduction from $L_D$ to $A_{TM}$. Thus $L_D \leq_M A_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in A_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{L}_M$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{L}_M$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in A_{TM}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin A_{TM}$. We consider two cases:

Case 1: $w$ is not a TM encoding. Then $f(w) = \langle S, \epsilon \rangle$. Since $S$ always accepts, $\epsilon \in \mathcal{L}_S$, and so $f(w) \notin A_{TM}$.

Case 2: $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in \mathcal{L}_M$. Then $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \in \mathcal{L}_M$, $f(w) \notin A_{TM}$.

In either case we have that $f(w) \notin A_{TM}$, as required. ■
Theorem: $L_D \leq_M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$.
Theorem: $L_D \leq_M \overline{A_{TM}}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A_{TM}}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$ if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ if and only if $f(w) \in A_{TM}$, then $f$ is a mapping reduction from $L_D$ to $A_{TM}$. Thus $L_D \leq M \overline{A_{TM}}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in A_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin (\mathcal{L}_M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin (\mathcal{L}_M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in A_{TM}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin A_{TM}$. We consider two cases:

Case 1: $w$ is not a TM encoding. Then $f(w) = \langle S, \varepsilon \rangle$. Since $S$ always accepts, $\varepsilon \in (\mathcal{L}_S)$, and so $f(w) \notin A_{TM}$.

Case 2: $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in (\mathcal{L}_M)$. Then $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \in (\mathcal{L}_M)$, $f(w) \notin A_{TM}$.

In either case we have that $f(w) \notin A_{TM}$, as required. ■
Theorem: $L_D \leq_M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$f(w) = \langle M, \langle M \rangle \rangle$ if $w$ is an encoding of a TM $M$.

$f(w) = \langle S, \varepsilon \rangle$ otherwise, where $S$ is a TM that always accepts.
**Theorem:** \( L_D \leq_M \overline{A}_{TM} \).

**Proof:** We exhibit a mapping reduction \( f \) from \( L_D \) to \( \overline{A}_{TM} \). Consider the function \( f \) defined as follows:

\[
f(w) = \langle M, \langle M \rangle \rangle \text{ if } w \text{ is an encoding of a TM } M.
\]
\[
f(w) = \langle S, \varepsilon \rangle \text{ otherwise, where } S \text{ is a TM that always accepts.}
\]

We claim that \( f \) can be computed by a TM and omit the details from this proof.

First, suppose that \( w \in L_D \). We prove that \( f(w) \in A_{TM} \). Since \( w \in L_D \), then it must have the form \( \langle M \rangle \) for some TM \( M \) such that \( \langle M \rangle \notin (\mathcal{L}_M) \).

Consequently, \( f(w) = \langle M, \langle M \rangle \rangle \). Since \( \langle M \rangle \notin (\mathcal{L}_M) \), we thus have that \( f(w) = \langle M, \langle M \rangle \rangle \in A_{TM} \).

Next, suppose that \( w \notin L_D \). We will prove that \( f(w) \notin A_{TM} \). We consider two cases:

**Case 1:** \( w \) is not a TM encoding. Then \( f(w) = \langle S, \varepsilon \rangle \). Since \( S \) always accepts, \( \varepsilon \in (\mathcal{L}_S) \), and so \( f(w) \notin A_{TM} \).

**Case 2:** \( w = \langle M \rangle \) for some TM \( M \) where \( \langle M \rangle \in (\mathcal{L}_M) \). Then \( f(w) = \langle M, \langle M \rangle \rangle \). Since \( \langle M \rangle \in (\mathcal{L}_M) \), \( f(w) \notin A_{TM} \).

In either case we have that \( f(w) \notin A_{TM} \), as required. ■
Theorem: \( \overline{L_D} \leq_M \overline{A_{TM}} \).

Proof: We exhibit a mapping reduction \( f \) from \( L_D \) to \( \overline{A_{TM}} \). Consider the function \( f \) defined as follows:

\[
\begin{align*}
  f(w) &= \langle M, \langle M \rangle \rangle \text{ if } w \text{ is an encoding of a TM } M. \\
  f(w) &= \langle S, \varepsilon \rangle \text{ otherwise, where } S \text{ is a TM that always accepts.}
\end{align*}
\]

We claim that \( f \) can be computed by a TM and omit the details from this proof. If we show that \( w \in L_D \) iff \( f(w) \in \overline{A_{TM}} \), then \( f \) is a mapping reduction from \( L_D \) to \( \overline{A_{TM}} \).
Theorem: $L_D \leq_M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$f(w) = \langle M, \langle M \rangle \rangle$ if $w$ is an encoding of a TM $M$.

$f(w) = \langle S, \varepsilon \rangle$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq_M \overline{A}_{TM}$. ■
Theorem: $L_D \leq^M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$ if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq^M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. 

...
Theorem: $L_D \leq_M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$ if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq_M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A}_{TM}$.
Theorem: \( L_D \leq_M A_{TM'} \).

Proof: We exhibit a mapping reduction \( f \) from \( L_D \) to \( A_{TM'} \). Consider the function \( f \) defined as follows:

\[
\begin{align*}
    f(w) &= \langle M, \langle M \rangle \rangle \text{ if } w \text{ is an encoding of a TM } M, \\
    f(w) &= \langle S, \varepsilon \rangle \text{ otherwise, where } S \text{ is a TM that always accepts.}
\end{align*}
\]

We claim that \( f \) can be computed by a TM and omit the details from this proof. If we show that \( w \in L_D \) iff \( f(w) \in A_{TM'} \), then \( f \) is a mapping reduction from \( L_D \) to \( A_{TM'} \). Thus \( L_D \leq_M A_{TM'} \).

First, suppose that \( w \in L_D \). We prove that \( f(w) \in A_{TM'} \). Since \( w \in L_D \), then it must have the form \( \langle M \rangle \) for some TM \( M \) such that \( \langle M \rangle \notin A(M) \).
**Theorem:** $L_D \leq_M \overline{A}_{TM}$.

**Proof:** We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$ if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq_M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A}_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. 


Theorem: \( L_D \leq_M \overline{A}_{TM} \).

Proof: We exhibit a mapping reduction \( f \) from \( L_D \) to \( \overline{A}_{TM} \). Consider the function \( f \) defined as follows:

\[
f(w) = \langle M, \langle M \rangle \rangle \text{ if } w \text{ is an encoding of a TM } M.
\]

\[
f(w) = \langle S, \varepsilon \rangle \text{ otherwise, where } S \text{ is a TM that always accepts.}
\]

We claim that \( f \) can be computed by a TM and omit the details from this proof. If we show that \( w \in L_D \) iff \( f(w) \in \overline{A}_{TM} \), then \( f \) is a mapping reduction from \( L_D \) to \( \overline{A}_{TM} \). Thus \( L_D \leq_M \overline{A}_{TM} \).

First, suppose that \( w \in L_D \). We prove that \( f(w) \in \overline{A}_{TM} \). Since \( w \in L_D \), then it must have the form \( \langle M \rangle \) for some TM \( M \) such that \( \langle M \rangle \notin A(M) \). Consequently, \( f(w) = \langle M, \langle M \rangle \rangle \). Since \( \langle M \rangle \notin A(M) \), we thus have that \( f(w) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM} \).
Theorem: $L_D \leq_M \overline{A_{TM}}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A_{TM}}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$

if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$

otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A_{TM}}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A_{TM}}$. Thus $L_D \leq_M \overline{A_{TM}}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A_{TM}}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{A}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A_{TM}}$.

Next, suppose that $w \notin L_D$. 
Theorem: $L_D \leq^M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$ if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq^M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A}_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{A}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin \overline{A}_{TM}$. 

\[\]
Theorem: $L_D \leq_M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

- $f(w) = \langle M, \langle M \rangle \rangle$ if $w$ is an encoding of a TM $M$.
- $f(w) = \langle S, \varepsilon \rangle$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq_M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A}_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{A}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin \overline{A}_{TM}$. We consider two cases:
Theorem: $L_D \leq_M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$ if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq_M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A}_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{A}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin \overline{A}_{TM}$. We consider two cases:

Case 1: $w$ is not a TM encoding.

Case 2: $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in \mathcal{A}(M)$.
**Theorem:** $L_D \leq_M \overline{A_{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A_{TM}}$. Consider the function $f$ defined as follows:

- $f(w) = \langle M, \langle M \rangle \rangle$ if $w$ is an encoding of a TM $M$.
- $f(w) = \langle S, \varepsilon \rangle$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A_{TM}}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A_{TM}}$. Thus $L_D \leq_M \overline{A_{TM}}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A_{TM}}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin A(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin A(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A_{TM}}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin \overline{A_{TM}}$. We consider two cases:

*Case 1:* $w$ is not a TM encoding. Then $f(w) = \langle S, \varepsilon \rangle$.

*Case 2:* $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in A(M)$.
Theorem: $L_D \leq_M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle$$ if $w$ is an encoding of a TM $M$.

$$f(w) = \langle S, \varepsilon \rangle$$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq_M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A}_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{A}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin \overline{A}_{TM}$. We consider two cases:

Case 1: $w$ is not a TM encoding. Then $f(w) = \langle S, \varepsilon \rangle$. Since $S$ always accepts, $\varepsilon \in \mathcal{A}(S)$, and so $f(w) \notin \overline{A}_{TM}$.

Case 2: $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in \mathcal{A}(M)$. 
**Theorem:** \( L_D \leq_M \overline{A}_{TM} \).

**Proof:** We exhibit a mapping reduction \( f \) from \( L_D \) to \( \overline{A}_{TM} \). Consider the function \( f \) defined as follows:

\[
 f(w) = \langle M, \langle M \rangle \rangle \text{ if } w \text{ is an encoding of a TM } M.
\]

\[
 f(w) = \langle S, \varepsilon \rangle \text{ otherwise, where } S \text{ is a TM that always accepts.}
\]

We claim that \( f \) can be computed by a TM and omit the details from this proof. If we show that \( w \in L_D \) iff \( f(w) \in \overline{A}_{TM} \), then \( f \) is a mapping reduction from \( L_D \) to \( \overline{A}_{TM} \). Thus \( L_D \leq_M \overline{A}_{TM} \).

First, suppose that \( w \in L_D \). We prove that \( f(w) \in \overline{A}_{TM} \). Since \( w \in L_D \), then it must have the form \( \langle M \rangle \) for some TM \( M \) such that \( \langle M \rangle \notin \mathcal{A}(M) \). Consequently, \( f(w) = \langle M, \langle M \rangle \rangle \). Since \( \langle M \rangle \notin \mathcal{A}(M) \), we thus have that \( f(w) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM} \).

Next, suppose that \( w \notin L_D \). We will prove that \( f(w) \notin \overline{A}_{TM} \). We consider two cases:

**Case 1:** \( w \) is not a TM encoding. Then \( f(w) = \langle S, \varepsilon \rangle \). Since \( S \) always accepts, \( \varepsilon \in \mathcal{A}(S) \), and so \( f(w) \notin \overline{A}_{TM} \).

**Case 2:** \( w = \langle M \rangle \) for some TM \( M \) where \( \langle M \rangle \in \mathcal{A}(M) \). Then \( f(w) = \langle M, \langle M \rangle \rangle \).
Theorem: $L_D \leq_R A_{\text{TM}}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $A_{\text{TM}}$. Consider the function $f$ defined as follows:

\[
f(w) = \langle M, \langle M \rangle \rangle \text{ if } w \text{ is an encoding of a TM } M.
\]
\[
f(w) = \langle S, \varepsilon \rangle \text{ otherwise, where } S \text{ is a TM that always accepts.}
\]

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in A_{\text{TM}}$, then $f$ is a mapping reduction from $L_D$ to $A_{\text{TM}}$. Thus $L_D \leq_R A_{\text{TM}}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in A_{\text{TM}}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{A}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in A_{\text{TM}}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin A_{\text{TM}}$. We consider two cases:

Case 1: $w$ is not a TM encoding. Then $f(w) = \langle S, \varepsilon \rangle$. Since $S$ always accepts, $\varepsilon \in \mathcal{A}(S)$, and so $f(w) \notin A_{\text{TM}}$.

Case 2: $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in \mathcal{A}(M)$. Then $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \in \mathcal{A}(M)$, $f(w) \notin A_{\text{TM}}$. 

■
Theorem: $L_D \leq^M \overline{A}_{TM}$.

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A}_{TM}$. Consider the function $f$ defined as follows:

$f(w) = \langle M, \langle M \rangle \rangle$ if $w$ is an encoding of a TM $M$.
$f(w) = \langle S, \varepsilon \rangle$ otherwise, where $S$ is a TM that always accepts.

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A}_{TM}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A}_{TM}$. Thus $L_D \leq^M \overline{A}_{TM}$.

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A}_{TM}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{A}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{A}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin \overline{A}_{TM}$. We consider two cases:

Case 1: $w$ is not a TM encoding. Then $f(w) = \langle S, \varepsilon \rangle$. Since $S$ always accepts, $\varepsilon \in \mathcal{A}(S)$, and so $f(w) \notin \overline{A}_{TM}$.

Case 2: $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in \mathcal{A}(M)$. Then $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \in \mathcal{A}(M)$, $f(w) \notin \overline{A}_{TM}$.

In either case we have that $f(w) \notin \overline{A}_{TM}$, as required.
Theorem: $L_D \leq_M \overline{A_{\text{TM}}}$.  

Proof: We exhibit a mapping reduction $f$ from $L_D$ to $\overline{A_{\text{TM}}}$. Consider the function $f$ defined as follows:

$$f(w) = \langle M, \langle M \rangle \rangle \text{ if } w \text{ is an encoding of a TM } M.$$  

$$f(w) = \langle S, \varepsilon \rangle \text{ otherwise, where } S \text{ is a TM that always accepts.}$$

We claim that $f$ can be computed by a TM and omit the details from this proof. If we show that $w \in L_D$ iff $f(w) \in \overline{A_{\text{TM}}}$, then $f$ is a mapping reduction from $L_D$ to $\overline{A_{\text{TM}}}$. Thus $L_D \leq_M \overline{A_{\text{TM}}}$.  

First, suppose that $w \in L_D$. We prove that $f(w) \in \overline{A_{\text{TM}}}$. Since $w \in L_D$, then it must have the form $\langle M \rangle$ for some TM $M$ such that $\langle M \rangle \notin \mathcal{I}(M)$. Consequently, $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \notin \mathcal{I}(M)$, we thus have that $f(w) = \langle M, \langle M \rangle \rangle \in \overline{A_{\text{TM}}}$.

Next, suppose that $w \notin L_D$. We will prove that $f(w) \notin \overline{A_{\text{TM}}}$. We consider two cases:

**Case 1:** $w$ is not a TM encoding. Then $f(w) = \langle S, \varepsilon \rangle$. Since $S$ always accepts, $\varepsilon \in \mathcal{I}(S)$, and so $f(w) \notin \overline{A_{\text{TM}}}$.

**Case 2:** $w = \langle M \rangle$ for some TM $M$ where $\langle M \rangle \in \mathcal{I}(M)$. Then $f(w) = \langle M, \langle M \rangle \rangle$. Since $\langle M \rangle \in \mathcal{I}(M)$, $f(w) \notin \overline{A_{\text{TM}}}$.

In either case we have that $f(w) \notin \overline{A_{\text{TM}}}$, as required.  ■
Simplifying our Encodings

- One of the challenges with the previous proof was the fact that we had to consider strings that didn't encode valid TMs.
- To make the math a bit easier, we'll assume that our encoding system is such that all strings encode valid objects.
- One option: Just say that every string that doesn't immediately encode a TM now encodes a TM that always rejects.
- As long as we're consistent, this is fine.
- This *dramatically* simplifies our proofs.
**Theorem:** \( L_D \leq_M \overline{A}_{TM} \).

**Proof:** We exhibit a mapping reduction \( f \) from \( L_D \) to \( \overline{A}_{TM} \). Consider the function \( f \) defined as follows:

\[
f(\langle M \rangle) = \langle M, \langle M \rangle \rangle
\]

We claim that \( f \) can be computed by a TM and omit the details from this proof. If we show that \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \overline{A}_{TM} \), then \( f \) is a mapping reduction from \( L_D \) to \( \overline{A}_{TM} \). Thus \( L_D \leq_M \overline{A}_{TM} \).

First, suppose that \( \langle M \rangle \in L_D \). We prove that \( f(\langle M \rangle) \in \overline{A}_{TM} \). Since \( \langle M \rangle \in L_D \), we know that \( \langle M \rangle \notin \mathcal{A}(M) \). Consequently, \( f(\langle M \rangle) = \langle M, \langle M \rangle \rangle \). Since \( \langle M \rangle \notin \mathcal{A}(M) \), we thus have that \( f(\langle M \rangle) = \langle M, \langle M \rangle \rangle \in \overline{A}_{TM} \).

Next, suppose that \( \langle M \rangle \notin L_D \). We will prove that \( f(\langle M \rangle) \notin \overline{A}_{TM} \). Since \( \langle M \rangle \notin L_D \), we must have that \( \langle M \rangle \in \mathcal{A}(M) \). Now, \( f(\langle M \rangle) = \langle M, \langle M \rangle \rangle \) and since \( \langle M \rangle \in \mathcal{A}(M) \), this means that \( f(\langle M \rangle) = \langle M, \langle M \rangle \rangle \notin L_D \), as required. \( \blacksquare \)
The Halting Problem

• Recall the definition of $HALT$: 

$$HALT = \{ \langle M, w \rangle \mid M \text{ halts on } w \}$$

• That is, the set of TM/string pairs where the TM $M$ either accepts or rejects the string $w$.

• Last time, we proved that $HALT \in \mathbb{RE} - \mathbb{R}$ by using reductions.

• Let's revisit these proofs in more detail, focusing on how reductions enable us to conclude this.
HALT is RE

- Recall: $A_{TM} \in RE$.

- To prove that HALT is RE, we will show that $HALT \leq_{M} A_{TM}$.

- Since $A_{TM} \in RE$, this proves that $HALT \in RE$.

- Idea: we need to find some function $f$ such that $\langle M, w \rangle \in HALT$ iff $f(\langle M, w \rangle) \in A_{TM}$.

  - Again, we assume that every string encodes a TM/string pair; if not, assume it encodes the machine that always rejects and $\varepsilon$. 
Goal: Choose our function $f(w)$ such that this machine $H$ is a recognizer for HALT.
\[ \text{HALT} \leq_{\text{TM}} A_{\text{TM}} \]

- Goal: Find a function \( f \) such that
  \[ \langle M, w \rangle \in \text{HALT} \iff f(\langle M, w \rangle) \in A_{\text{TM}} \]
- Substituting the definitions:
  \[ M \text{ halts on } w \iff f(\langle M, w \rangle) \in A_{\text{TM}}. \]
- Assume that \( f(\langle M, w \rangle) = \langle M', w' \rangle \) for some TM \( M' \) and string \( w' \). Then we have
  \[ M \text{ halts on } w \iff \langle M', w' \rangle \in A_{\text{TM}} \]
  \[ M \text{ halts on } w \iff w' \in L(M') \]
  \[ M \text{ halts on } w \iff M' \text{ accepts } w \]
Choosing $M'$ and $w'$

- We need to find $M'$ and $w'$ such that $M$ halts on $w$ iff $M'$ accepts $w'$.
- This is the creative step of the proof – how do we choose an $M'$ and $w'$ with that property?
- **Key idea that shows up in almost all major reduction proofs**: Construct a machine $M'$ and string $w'$ so that running $M'$ on $w'$ runs $M$ on $w$.
- This causes the behavior of $M'$ running on $w'$ to depend on what $M$ does on $w$. 
Choosing $M'$ and $w'$

Here is one possible choice of $M'$ and $w'$ we can make:

$$M' = \text{“On input } \langle N, z \rangle:$$

- Run $N$ on $z$.
- If $N$ halts on $z$, accept.

$$w' = \langle M, w \rangle$$

Now, running $M'$ on $w'$ runs $M$ on $w$. If $M$ halts on $w$, then $M'$ accepts $w'$. If $M$ loops on $w$, then $M'$ does not accept $w'$. 
Compute \( f \)

Machine \( H \) accepts \( \langle M, w \rangle \) iff \( R \) accepts \( \langle M', \langle M, w \rangle \rangle \) iff \( \langle M', \langle M, w \rangle \rangle \in A_{TM} \) iff \( M' \) accepts \( \langle M, w \rangle \) iff \( M \) halts on \( w \) iff \( \langle M, w \rangle \in HALT \).

\[ H = \text{“On input } \langle M, w \rangle \text{: Compute } \langle M', \langle M, w \rangle \rangle \text{. Run } R \text{ on } \langle M', \langle M, w \rangle \rangle \text{. If } R \text{ accepts } \langle M', \langle M, w \rangle \rangle \text{, accept } \langle M, w \rangle \text{. If } R \text{ rejects } \langle M', \langle M, w \rangle \rangle \text{, reject } \langle M, w \rangle \text{.”} \]
Theorem: \( \text{HALT} \leq_{M} A_{\text{TM}} \).
Theorem: $\text{HALT} \leq^M A_{\text{TM}}$.

Proof: We exhibit a mapping reduction $f$ from $\text{HALT}$ to $A_{\text{TM}}$.

Let the machine $M'$ be defined as follows:

$M' = \text{On input } \langle N, z \rangle: \text{Run } N \text{ on } z. \text{If } N \text{ halts on } z, \text{ accept.}$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof.

We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in A_{\text{TM}}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in A_{\text{TM}}$ iff $M'$ accepts $\langle M, w \rangle$. By construction, $M'$ accepts $\langle M, w \rangle$ iff $M$ halts on $w$. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$.

Thus $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in A_{\text{TM}}$.

Therefore, $f$ is a mapping reduction from $\text{HALT}$ to $A_{\text{TM}}$, so $\text{HALT} \leq^M A_{\text{TM}}$. ■
**Theorem:** $\text{HALT} \leq_{M} A_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $\text{HALT}$ to $A_{\text{TM}}$. Let the machine $M'$ be defined as follows:

Let the machine $M'$ be defined as follows:
Theorem: $\text{HALT} \leq_m A_{\text{TM}}$.

Proof: We exhibit a mapping reduction $f$ from $\text{HALT}$ to $A_{\text{TM}}$. Let the machine $M'$ be defined as follows:

$M' = \text{"On input } \langle N, z \rangle:\n\quad \text{Run } N \text{ on } z.\
\quad \text{If } N \text{ halts on } z, \text{ accept."}$
Theorem: HALT $\leq_M A_{TM}$.

Proof: We exhibit a mapping reduction $f$ from HALT to $A_{TM}$. Let the machine $M'$ be defined as follows:

$$M' = \text{"On input } \langle N, z \rangle: \text{\ Run } N \text{ on } z. \text{\ If } N \text{ halts on } z, \text{ accept."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. 
**Theorem:** $\text{HALT} \leq^M A_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $\text{HALT}$ to $A_{\text{TM}}$. Let the machine $M'$ be defined as follows:

$$M' = \text{"On input } \langle N, z \rangle:\n\text{Run } N \text{ on } z.\n\text{If } N \text{ halts on } z, \text{ accept."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof.
**Theorem:** $HALT \leq_{M} A_{TM}$.

**Proof:** We exhibit a mapping reduction $f$ from $HALT$ to $A_{TM}$.

Let the machine $M'$ be defined as follows:

\[
M' = \text{"On input } \langle N, z \rangle:\ \\
\text{Run } N \text{ on } z. \\
\text{If } N \text{ halts on } z, \text{ accept."}
\]

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in HALT$ iff $f(\langle M, w \rangle) \in A_{TM}$. 
**Theorem:** $\text{HALT} \leq_{M} \text{A}_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $\text{HALT}$ to $\text{A}_{\text{TM}}$. Let the machine $M'$ be defined as follows:

$$M' = \text{"On input } \langle N, z \rangle:\n\quad \text{Run } N \text{ on } z.\n\quad \text{If } N \text{ halts on } z, \text{ accept."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{A}_{\text{TM}}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in \text{A}_{\text{TM}}$ iff $M'$ accepts $\langle M, w \rangle$. 


**Theorem:** $\text{HALT} \leq^M \text{A}_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $\text{HALT}$ to $\text{A}_{\text{TM}}$.

Let the machine $M'$ be defined as follows:

$M' = \text{"On input $\langle N, z \rangle$:}\\
\quad \text{Run } N \text{ on } z.\\
\quad \text{If } N \text{ halts on } z, \text{ accept."}$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{A}_{\text{TM}}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in \text{A}_{\text{TM}}$ iff $M'$ accepts $\langle M, w \rangle$. By construction, $M'$ accepts $\langle M, w \rangle$ iff $M$ halts on $w$. 


**Theorem:** $\text{HALT} \leq_M A_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $\text{HALT}$ to $A_{\text{TM}}$.

Let the machine $M'$ be defined as follows:

$$M' = \text{“On input } \langle N, z \rangle\text{:}
\begin{align*}
\text{Run } N \text{ on } z. \\
\text{If } N \text{ halts on } z, \text{ accept.”}
\end{align*}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in A_{\text{TM}}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in A_{\text{TM}}$ iff $M'$ accepts $\langle M, w \rangle$. By construction, $M'$ accepts $\langle M, w \rangle$ iff $M$ halts on $w$. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$.
**Theorem:** $\text{HALT} \leq_m A_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $\text{HALT}$ to $A_{\text{TM}}$. Let the machine $M'$ be defined as follows:

$$M' = \text{"On input } \langle N, z \rangle:\$$
$$\quad \text{Run } N \text{ on } z.$$
$$\quad \text{If } N \text{ halts on } z, \text{ accept."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in A_{\text{TM}}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in A_{\text{TM}}$ iff $M'$ accepts $\langle M, w \rangle$. By construction, $M'$ accepts $\langle M, w \rangle$ iff $M$ halts on $w$. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$. Thus $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in A_{\text{TM}}$. ■
**Theorem:** $\text{HALT} \leq_{M} A_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction $f$ from $\text{HALT}$ to $A_{\text{TM}}$. Let the machine $M'$ be defined as follows:

\[ M' = \text{"On input } \langle N, z \rangle:\]
\[ \text{Run } N \text{ on } z. \]
\[ \text{If } N \text{ halts on } z, \text{ accept."} \]

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in A_{\text{TM}}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in A_{\text{TM}}$ iff $M'$ accepts $\langle M, w \rangle$. By construction, $M'$ accepts $\langle M, w \rangle$ iff $M$ halts on $w$. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$. Thus $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in A_{\text{TM}}$. Therefore, $f$ is a mapping reduction from $\text{HALT}$ to $A_{\text{TM}}$, so $\text{HALT} \leq_{M} A_{\text{TM}}$. ■
**Theorem:** \(\text{HALT} \leq_{M} \text{A}_{\text{TM}}.\)

**Proof:** We exhibit a mapping reduction \(f\) from \(\text{HALT}\) to \(\text{A}_{\text{TM}}\). Let the machine \(M'\) be defined as follows:

\[
M' = \text{"On input } \langle N, z \rangle:\n\quad \text{Run } N \text{ on } z.
\quad \text{If } N \text{ halts on } z, \text{ accept."
}\]

Then let \(f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle\). We claim that \(f\) is computable and omit the details from this proof. We further claim that \(\langle M, w \rangle \in \text{HALT}\) iff \(f(\langle M, w \rangle) \in \text{A}_{\text{TM}}\). To see this, note that \(f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in \text{A}_{\text{TM}}\) iff \(M'\) accepts \(\langle M, w \rangle\). By construction, \(M'\) accepts \(\langle M, w \rangle\) iff \(M\) halts on \(w\). Finally, note that \(M\) halts on \(w\) iff \(\langle M, w \rangle \in \text{HALT}\). Thus \(\langle M, w \rangle \in \text{HALT}\) iff \(f(\langle M, w \rangle) \in \text{A}_{\text{TM}}\). Therefore, \(f\) is a mapping reduction from \(\text{HALT}\) to \(\text{A}_{\text{TM}}\), so \(\text{HALT} \leq_{M} \text{A}_{\text{TM}}.\) ■
HALT is Undecidable

- We proved $\text{HALT} \in \text{RE}$ by showing that $\text{HALT} \leq_{m} A_{TM}$.
- We can prove $\text{HALT} \notin \text{R}$ by showing that $A_{TM} \leq_{m} \text{HALT}$.
- Note that this has to be a completely separate reduction! We're transforming $A_{TM}$ into $\text{HALT}$ this time, not the other way around.
We want to find a computable function $f$ such that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in HALT$.

Assume $f(\langle M, w \rangle)$ has the form $\langle M', w' \rangle$ for some TM $M'$ and string $w'$.

We want $\langle M, w \rangle \in A_{TM}$ iff $\langle M', w' \rangle \in HALT$.

Substituting definitions:

$M$ accepts $w$ iff $M'$ halts on $w'$.

How might we design $M'$ and $w'$?
\[ A_{TM} \leq^M HALT \]

- We need to choose a TM/string pair \( M' \) and \( w' \) such that \( M' \) halts on \( w' \) iff \( M \) accepts \( w \).
- Repeated idea: Construct \( M' \) and \( w' \) such that running \( M' \) on \( w' \) simulates \( M \) on \( w \) and bases its decision on what happens.
- One option:
  
  \[
  M' = \text{“On input } \langle N, z \rangle \text{:} \\
  \text{Run } N \text{ on } z. \\
  \text{If } N \text{ accepts } z, \text{ accept.} \\
  \text{If } N \text{ rejects } z, \text{ loop infinitely.”} \\
  \]
  
  \[
  w' = \langle M, w \rangle
  \]
Machine $H$ accepts $\langle M, w \rangle$ iff $R$ accepts $\langle M', \langle M, w \rangle \rangle$ iff $\langle M', \langle M, w \rangle \rangle \in \text{HALT}$ iff $M'$ halts on $\langle M, w \rangle$ iff $M$ accepts $w$ iff $\langle M, w \rangle \in A_{\text{TM}}$.

$H = \text{``On input } \langle M, w \rangle:\n\quad \text{Compute } \langle M', \langle M, w \rangle \rangle.\n\quad \text{Run } R \text{ on } \langle M', \langle M, w \rangle \rangle.\n\quad \text{If } R \text{ accepts } \langle M', \langle M, w \rangle \rangle, \text{ accept } \langle M, w \rangle.\n\quad \text{If } R \text{ rejects } \langle M', \langle M, w \rangle \rangle, \text{ reject } \langle M, w \rangle\text{.''}$
Theorem: \( A_{\text{TM}} \leq_{M} \text{HALT} \).
Theorem: $A_{TM} \leq_M HALT$.
Proof: We exhibit a mapping reduction from $A_{TM}$ to $HALT$. 
Theorem: $A_{TM} \leq_{M} HALT$.

Proof: We exhibit a mapping reduction from $A_{TM}$ to $HALT$. Let $M'$ be the following TM:

Let $M'$ be the following TM:
Theorem: $A_{TM} \leq_M HALT$.

Proof: We exhibit a mapping reduction from $A_{TM}$ to $HALT$. Let $M'$ be the following TM:

$$M' = \text{"On input } \langle N, z \rangle:\text{ Run } N \text{ on } z.$$
$$\text{If } N \text{ accepts, accept.}$$
$$\text{If } N \text{ rejects, loop infinitely."}$$
Theorem: $A_{TM} \leq_{M} \text{HALT}$. 

Proof: We exhibit a mapping reduction from $A_{TM}$ to $\text{HALT}$. Let $M'$ be the following TM:

$$M' = \text{"On input } \langle N, z \rangle:\text{ Run } N \text{ on } z.\text{ If } N \text{ accepts, accept. If } N \text{ rejects, loop infinitely."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. 
Theorem: $A_{TM} \leq^M HALT$.

Proof: We exhibit a mapping reduction from $A_{TM}$ to $HALT$. Let $M'$ be the following TM:

\[ M' = \text{"On input } \langle N, z \rangle:\]
\[ \quad \text{Run } N \text{ on } z. \]
\[ \quad \text{If } N \text{ accepts, accept.} \]
\[ \quad \text{If } N \text{ rejects, loop infinitely."} \]

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof.
**Theorem:** $A_{TM} \leq_M HALT$.

**Proof:** We exhibit a mapping reduction from $A_{TM}$ to $HALT$. Let $M'$ be the following TM:

$$M' = \text{"On input } \langle N, z \rangle:\$$
  - Run $N$ on $z$.
  - If $N$ accepts, accept.
  - If $N$ rejects, loop infinitely.”

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in HALT$. 
Theorem: $A_{\text{TM}} \leq_M \text{HALT}$.

Proof: We exhibit a mapping reduction from $A_{\text{TM}}$ to $\text{HALT}$. Let $M'$ be the following TM:

$$M' = \text{"On input } \langle N, z \rangle:\text{ Run } N \text{ on } z.\text{ If } N \text{ accepts, accept. If } N \text{ rejects, loop infinitely."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in A_{\text{TM}}$ iff $f(\langle M, w \rangle) \in \text{HALT}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in \text{HALT}$ iff $M'$ halts on $\langle M, w \rangle$. 
**Theorem:** \( \mathsf{A}_{\mathsf{TM}} \leq_{\mathsf{M}} \mathsf{HALT} \).

**Proof:** We exhibit a mapping reduction from \( \mathsf{A}_{\mathsf{TM}} \) to \( \mathsf{HALT} \).

Let \( M' \) be the following TM:

\[
M' = \text{"On input } \langle N, z \rangle:\
\quad \text{Run } N \text{ on } z.
\quad \text{If } N \text{ accepts, accept.}
\quad \text{If } N \text{ rejects, loop infinitely."
}\]

Then let \( f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \). We claim that \( f \) is computable and omit the details from this proof. We further claim that \( \langle M, w \rangle \in \mathsf{A}_{\mathsf{TM}} \) iff \( f(\langle M, w \rangle) \in \mathsf{HALT} \). To see this, note that \( f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in \mathsf{HALT} \) iff \( M' \) halts on \( \langle M, w \rangle \). By construction, \( M' \) halts on \( \langle M, w \rangle \) iff \( M \) accepts \( w \).
Theorem: $A_{TM} \leq_M HALT$.

Proof: We exhibit a mapping reduction from $A_{TM}$ to $HALT$. Let $M'$ be the following TM:

$$M' = "On input \langle N, z \rangle:
\begin{array}{l}
\text{Run } N \text{ on } z.
\text{If } N \text{ accepts, accept.}
\text{If } N \text{ rejects, loop infinitely."
}\end{array}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in HALT$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in HALT$ iff $M'$ halts on $\langle M, w \rangle$. By construction, $M'$ halts on $\langle M, w \rangle$ iff $M$ accepts $w$. Finally, $M$ accepts $w$ iff $\langle M, w \rangle \in A_{TM}$. ■
Theorem: $A_{TM} \leq_M HALT$.

Proof: We exhibit a mapping reduction from $A_{TM}$ to $HALT$. Let $M'$ be the following TM:

$$M' = \text{“On input } \langle N, z \rangle:\n\text{Run } N \text{ on } z.\n\text{If } N \text{ accepts, accept.}\n\text{If } N \text{ rejects, loop infinitely.”}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in HALT$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in HALT$ iff $M'$ halts on $\langle M, w \rangle$. By construction, $M'$ halts on $\langle M, w \rangle$ iff $M$ accepts $w$. Finally, $M$ accepts $w$ iff $\langle M, w \rangle \in A_{TM}$. Thus we have that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in HALT$. ■
Theorem: $A_{\text{TM}} \leq_{M} \text{HALT}$.

Proof: We exhibit a mapping reduction from $A_{\text{TM}}$ to $\text{HALT}$. Let $M'$ be the following TM:

$$M' = \text{"On input } \langle N, z \rangle:\n\text{Run } N \text{ on } z.\n\text{If } N \text{ accepts, accept.}\n\text{If } N \text{ rejects, loop infinitely."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in A_{\text{TM}}$ iff $f(\langle M, w \rangle) \in \text{HALT}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in \text{HALT}$ iff $M'$ halts on $\langle M, w \rangle$. By construction, $M'$ halts on $\langle M, w \rangle$ iff $M$ accepts $w$. Finally, $M$ accepts $w$ iff $\langle M, w \rangle \in A_{\text{TM}}$. Thus we have that $\langle M, w \rangle \in A_{\text{TM}}$ iff $f(\langle M, w \rangle) \in \text{HALT}$. Therefore, $f$ is a mapping reduction from $A_{\text{TM}}$ to $\text{HALT}$, so $A_{\text{TM}} \leq_{M} \text{HALT}$. ■
Theorem: $A_{\text{TM}} \leq_{M} \text{HALT}$.

Proof: We exhibit a mapping reduction from $A_{\text{TM}}$ to HALT.

Let $M'$ be the following TM:

$$M' = \text{"On input } \langle N, z \rangle: \text{ Run } N \text{ on } z. \text{ If } N \text{ accepts, accept. If } N \text{ rejects, loop infinitely."}$$

Then let $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle$. We claim that $f$ is computable and omit the details from this proof. We further claim that $\langle M, w \rangle \in A_{\text{TM}}$ iff $f(\langle M, w \rangle) \in \text{HALT}$. To see this, note that $f(\langle M, w \rangle) = \langle M', \langle M, w \rangle \rangle \in \text{HALT}$ iff $M'$ halts on $\langle M, w \rangle$. By construction, $M'$ halts on $\langle M, w \rangle$ iff $M$ accepts $w$. Finally, $M$ accepts $w$ iff $\langle M, w \rangle \in A_{\text{TM}}$.

Thus we have that $\langle M, w \rangle \in A_{\text{TM}}$ iff $f(\langle M, w \rangle) \in \text{HALT}$. Therefore, $f$ is a mapping reduction from $A_{\text{TM}}$ to HALT, so $A_{\text{TM}} \leq_{M} \text{HALT}$. ■
What's Going On?

• The structure of our reduction is as follows:
  • Assume, for the sake of contradiction, that you can decide whether any machine is going to halt or not.
  • Then I could trick your machine into deciding $A_{TM}$ by constructing a machine that halts on a specific input precisely if a specific TM accepts a specific string.
  • Since your decider can decide $HALT$ for any TM/string pair, it should be able to decide it for this particular machine.
  • Your machine, in attempting to solve an entirely different problem, can be tricked into solving $A_{TM}$. 
An Important Detail

- In the course of this reduction, we constructed a new machine $M'$.  
- We never actually run the machine $M'$! That might loop forever.  
- We instead just build a description of that machine and fed it into our machine for $HALT$. 
- The answer given back by this machine about what $M'$ would do if we were to run it can then be used to solve $A_{TM}$.  

A More Elaborate Reduction

• We have just shown that we can't decide whether or not a TM halts on a specific input.

• It seems, therefore, that we shouldn't be able to decide whether a TM halts on all possible inputs.

• Consider the language

$$ DECIDER = \{ \langle M \rangle \mid M \text{ is a decider} \} $$

• How would we prove that $DECIDER$ is, itself, undecidable?
We will prove that DECIDER is undecidable by reducing HALT to DECIDER.

Want to find a function $f$ such that

$\langle M, w \rangle \in HALT \iff f(\langle M, w \rangle) \in DECIDER$.

Assuming that $f(\langle M, w \rangle) = \langle M' \rangle$ for some TM $M'$, we have that

$\langle M, w \rangle \in HALT \iff \langle M' \rangle \in DECIDER$.

$M$ halts on $w$ iff $M'$ is a decider.

$M$ halts on $w$ iff $M'$ halts on all inputs.
The Reduction

• Find a TM $M'$ such that $M'$ halts on all inputs iff $M$ halts on $w$.

• Key idea from before: Build $M'$ such that running $M'$ on any input runs $M$ on $w$.

• Here is one choice of $M'$:

  $M' = \text{“On input } x:\n\text{Ignore } x.\n\text{Run } M \text{ on } w.\n\text{If } M \text{ halts on } w, \text{ accept.”}$

• Notice that $M'$ “amplifies” what $M$ does on $w$:
  • If $M$ halts on $w$, $M'$ halts on every input.
  • If $M$ loops on $w$, $M'$ loops on every input.
Justifying $M'$

- Notice that our machine $M'$ has the machine $M$ and string $w$ built into it!
- This is different from the machines we have constructed in the past.
- How do we justify that it's possible for some TM to construct a new TM at all?

\[
M' = \text{“On input } x:\text{ Ignore } x.\text{ Run } M \text{ on } w. \text{ If } M \text{ halts on } w, \text{ accept.”}
\]
The Parameterization Theorem

**Theorem:** Let $M$ be a TM of the form

$M = \text{“On input } \langle x_1, x_2, \ldots, x_n \rangle:\text{ Do something with } x_1, x_2, \ldots, x_n \text{”}$

and any value $p$ for parameter $x_1$, then a TM can construct the following TM $M'$:

$M' = \text{“On input } \langle x_2, \ldots, x_n \rangle:\text{ Do something with } p, x_2, \ldots, x_n \text{”}$
Justifying $M'$

- Consider this machine $X$:
  
  $X = \text{"On input } \langle N, z, x \rangle:\"
  
  Ignore $x$.
  
  Run $N$ on $z$.
  
  If $N$ halts on $z$, accept."

- Applying the parameterization theorem twice with the values $M$ and $w$ produces the machine
  
  $X = \text{"On input } x:\"
  
  Ignore $x$.
  
  Run $M$ on $w$.
  
  If $M$ halts on $w$, accept."
$H = \text{"On input } \langle M, w \rangle \text{:} $
\begin{align*}
&\text{Compute } \langle M' \rangle. \\
&\text{Run } R \text{ on } \langle M' \rangle. \\
&\text{If } R \text{ accepts } \langle M' \rangle, \text{ accept } \langle M, w \rangle. \\
&\text{If } R \text{ rejects } \langle M' \rangle, \text{ reject } \langle M, w \rangle. 
\end{align*}
Theorem: \( \text{HALT} \leq_{\text{M}} \text{DECIDER} \).
Theorem: $\text{HALT} \leq_M \text{DECIDER}$.  
Proof: We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. 
Theorem: \( \text{HALT} \leq_{\text{M}} \text{DECIDER} \).

Proof: We exhibit a mapping reduction from \( \text{HALT} \) to \( \text{DECIDER} \).

For any TM/string pair \( \langle M, w \rangle \), let \( f(\langle M, w \rangle) = \langle M' \rangle \), where \( \langle M' \rangle \) is defined in terms of \( M \) and \( w \) as follows:
Theorem: \( \text{HALT} \leq_{M} \text{DECIDER} \).

Proof: We exhibit a mapping reduction from \( \text{HALT} \) to \( \text{DECIDER} \).

For any TM/string pair \( \langle M, w \rangle \), let \( f(\langle M, w \rangle) = \langle M' \rangle \), where \( \langle M' \rangle \) is defined in terms of \( M \) and \( w \) as follows:

\[
M' = \text{"On input } x: \\
\quad \text{Ignore } x. \\
\quad \text{Run } M \text{ on } w. \\
\quad \text{If } M \text{ halts on } w, \text{ accept."}
\]
Theorem: \( \text{HALT} \leq_m \text{DECIDER} \).

Proof: We exhibit a mapping reduction from \( \text{HALT} \) to \( \text{DECIDER} \). For any TM/string pair \( \langle M, w \rangle \), let \( f(\langle M, w \rangle) = \langle M' \rangle \), where \( \langle M' \rangle \) is defined in terms of \( M \) and \( w \) as follows:

\[
M' = \text{“On input } x:\n    \text{Ignore } x.\n    \text{Run } M \text{ on } w.\n    \text{If } M \text{ halts on } w, \text{ accept.”}
\]

By the parameterization theorem, \( f \) is a computable function.
Theorem: $\text{HALT} \leq_m \text{DECIDER}$.

Proof: We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

$$M' = "\text{On input } x:\n\quad \text{Ignore } x.\n\quad \text{Run } M \text{ on } w.\n\quad \text{If } M \text{ halts on } w, \text{ accept.}"$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. 
Theorem: \( \text{HALT} \leq^M \text{DECIDER} \).

Proof: We exhibit a mapping reduction from \( \text{HALT} \) to \( \text{DECIDER} \). For any TM/string pair \( \langle M, w \rangle \), let \( f(\langle M, w \rangle) = \langle M' \rangle \), where \( \langle M' \rangle \) is defined in terms of \( M \) and \( w \) as follows:

\[
M' = \text{"On input } x:\n\quad \text{Ignore } x. \\
\quad \text{Run } M \text{ on } w. \\
\quad \text{If } M \text{ halts on } w, \text{ accept."
}\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M, w \rangle \in \text{HALT} \) iff \( f(\langle M, w \rangle) \in \text{DECIDER} \). To see this, note that \( f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER} \) iff \( M' \) halts on all inputs.
**Theorem:** $\text{HALT} \leq_M \text{DECIDER}$.  

**Proof:** We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

$$M' = \text{"On input } x: \text{ Ignore } x. \text{ Run } M \text{ on } w. \text{ If } M \text{ halts on } w, \text{ accept."}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER}$ iff $M'$ halts on all inputs. We claim that $M'$ halts on all inputs iff $M$ halts on $w$. 


**Theorem:** $\text{HALT} \leq M \text{DECIDER}.$

**Proof:** We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

\[ M' = \text{“On input } x: \]
\[ \quad \text{Ignore } x. \]
\[ \quad \text{Run } M \text{ on } w. \]
\[ \quad \text{If } M \text{ halts on } w, \text{ accept.”} \]

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER}$ iff $M'$ halts on all inputs. We claim that $M'$ halts on all inputs iff $M$ halts on $w$. To see this, note that when $M'$ is run on any input, it halts iff $M$ halts on $w$. 


Theorem: $\text{HALT} \leq_m \text{DECIDER}$.

Proof: We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

$$M' = \text{"On input } x: \]
  \text{Ignore } x.
  \text{Run } M \text{ on } w.
  \text{If } M \text{ halts on } w, \text{ accept.}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER}$ iff $M'$ halts on all inputs. We claim that $M'$ halts on all inputs iff $M$ halts on $w$. To see this, note that when $M'$ is run on any input, it halts iff $M$ halts on $w$. Thus if $M$ halts on $w$, then $M'$ halts on all inputs, and if $M$ loops on $w$, $M'$ loops on all inputs.
Theorem: $\text{HALT} \leq^M \text{DECIDER}$.  

Proof: We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

$$M' = \text{"On input } x:\text{ Ignore } x.\text{ Run } M \text{ on } w.\text{ If } M \text{ halts on } w, \text{ accept."}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER}$ iff $M'$ halts on all inputs. We claim that $M'$ halts on all inputs iff $M$ halts on $w$. To see this, note that when $M'$ is run on any input, it halts iff $M$ halts on $w$. Thus if $M$ halts on $w$, then $M'$ halts on all inputs, and if $M$ loops on $w$, $M'$ loops on all inputs. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$.  


Theorem: $\text{HALT} \leq^M \text{DECIDER}$.

Proof: We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

$$M' = \text{“On input } x:\$$
  $$\text{Ignore } x.$$  
  $$\text{Run } M \text{ on } w.$$  
  $$\text{If } M \text{ halts on } w, \text{ accept.”}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER}$ iff $M'$ halts on all inputs. We claim that $M'$ halts on all inputs iff $M$ halts on $w$. To see this, note that when $M'$ is run on any input, it halts iff $M$ halts on $w$. Thus if $M$ halts on $w$, then $M'$ halts on all inputs, and if $M$ loops on $w$, $M'$ loops on all inputs. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$. Thus $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. ■
Theorem: $\text{HALT} \leq^m \text{DECIDER}$.

Proof: We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

$$M' = \text{“On input } x:\n\quad \text{Ignore } x.
\quad \text{Run } M \text{ on } w.
\quad \text{If } M \text{ halts on } w, \text{ accept.”}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER}$ iff $M'$ halts on all inputs. We claim that $M'$ halts on all inputs iff $M$ halts on $w$. To see this, note that when $M'$ is run on any input, it halts iff $M$ halts on $w$. Thus if $M$ halts on $w$, then $M'$ halts on all inputs, and if $M$ loops on $w$, $M'$ loops on all inputs. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$. Thus $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. Therefore, $f$ is a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$, so $\text{HALT} \leq^m \text{DECIDER}$. ■
Theorem: $\text{HALT} \leq_{M} \text{DECIDER}$.

Proof: We exhibit a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $\langle M' \rangle$ is defined in terms of $M$ and $w$ as follows:

$$M' = \text{"On input } x:\text{ Ignore } x.$$
$$\text{Run } M \text{ on } w.$$
$$\text{If } M \text{ halts on } w, \text{ accept."}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in \text{DECIDER}$ iff $M'$ halts on all inputs. We claim that $M'$ halts on all inputs iff $M$ halts on $w$. To see this, note that when $M'$ is run on any input, it halts iff $M$ halts on $w$. Thus if $M$ halts on $w$, then $M'$ halts on all inputs, and if $M$ loops on $w$, $M'$ loops on all inputs. Finally, note that $M$ halts on $w$ iff $\langle M, w \rangle \in \text{HALT}$. Thus $\langle M, w \rangle \in \text{HALT}$ iff $f(\langle M, w \rangle) \in \text{DECIDER}$. Therefore, $f$ is a mapping reduction from $\text{HALT}$ to $\text{DECIDER}$, so $\text{HALT} \leq_{M} \text{DECIDER}$. ■
Next Time

• More Reductions
  • What other types of machines can we build?

• co-RE
  • Are there unsolvable problems for which we have hope?