Reductions
Part Two
Mapping Reducibility

- A **mapping reduction** from $A$ to $B$ is a function $f$ such that
  - $f$ is computable, and
  - For any $w$, $w \in A$ iff $f(w) \in B$.
- 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 $B$. 
Why Mapping Reducibility Matters

If this one is “easy” ($R$ or $RE$)…

$A \leq_M B$

... 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

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

\[ H \text{ accepts } w \iff M \text{ accepts } f(w) \iff f(w) \in B \iff w \in A. \]
An Elaborate Reduction

• 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 $\textsc{Decider}$ is undecidable by reducing $\textsc{Halt}$ to $\textsc{Decider}$.

Want to find a function $f$ such that

$$\langle M, w \rangle \in \textsc{Halt} \iff f(\langle M, w \rangle) \in \textsc{Decider}.$$ 

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

$$\langle M, w \rangle \in \textsc{Halt} \iff \langle M' \rangle \in \textsc{Decider}.$$ 

$M$ 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: Build $M'$ such that running $M'$ on any input runs $M$ on $w$.
- Here is one choice of $M'$:
  
  $M' = \text{"On input } x:\n  \text{ Ignore } x.\n  \text{ Run } M \text{ on } w.\n  \text{ If } M \text{ halts on } w, \text{ accept. }$\n
- 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.
The Parameterization Theorem

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

\[ M = \text{“On input } \langle x_1, x_2, \ldots, x_n \rangle:\]

Do something with $x_1, x_2, \ldots, x_n$’

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

\[ M' = \text{“On input } \langle x_2, \ldots, x_n \rangle:\]

Do something with $p, x_2, \ldots, x_n$"
Justifying $M'$

- Consider this machine $X$:
  
  \[
  X = \text{“On input } \langle N, z, x \rangle:\n  \]
  
  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
  
  \[
  X = \text{“On input } x:\n  \]
  
  Ignore $x$.
  
  Run $M$ on $w$.
  
  If $M$ halts on $w$, accept.”
DECIDER is Undecidable

Decider for DECIDER
DECIDER is Undecidable

\[(M, w)\]
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

(Ignored)

(x)
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

Simulate $M$ on $w$

(Ignored)
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

Simulate $M$ on $w$

(Ignored)

Machine $M'$
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

$M' = \text{"On input } x: \text{ Ignore } x. \text{ Run } M \text{ on } w. \text{ If } M \text{ accepts } w, \text{ accept. If } M \text{ rejects } w, \text{ reject."}$
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

Simulate $M$ on $w$

(Ignored)

Machine $M'$

$x$
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

What does $M'$ do if $M$ halts on $w$?
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

What does $M'$ do if $M$ halts on $w$?

$M'$ always halts
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

What does $M'$ do if $M$ loops on $w$?
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

What does $M'$ do if $M$ loops on $w$?

$M'$ never halts
DECIDER is Undecidable

⟨M, w⟩ → Construct M' from ⟨M, w⟩
          | Decider for DECIDER
          +------------------+
          |                 |
          +------------------+

⟨M, w⟩ → Simulate M on w
          | (Ignored)
          +------------------+
          |                 |
          +------------------+

x → Machine M'
          | (Ignored)
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

$\langle M', w \rangle$

Decider for DECIDER

Simulate $M$ on $w$

$Ignored$

$x$

$Ignored$

Machine $M'$
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Simulate $M$ on $w$

$Ignored$

Machine $H$

Machine $M'$

Decider for DECIDER
DECIDER is Undecidable

Construct M' from ⟨M, w⟩

Decider for DECIDER

 ⟨M', w⟩

What does H do if M halts on w?
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

$\langle M' \rangle$

(Always Halts)

What does $H$ do if $M$ halts on $w$?

Machine $H$

Simulate $M$ on $w$

(Ignored)
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

$\langle M' \rangle$

Decider for DECIDER

What does $H$ do if $M$ halts on $w$?
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

$\langle M' \rangle$

Decider for DECIDER

Machine $H$

Simulate $M$ on $w$

(Ignored)
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

Machine H

Simulate $M$ on $w$

(ignored)

What does H do if $M$ loops on $w$?
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

$\langle M' \rangle$ (Never Halts)

Decider for DECIDER

What does $H$ do if $M$ loops on $w$?
DECIDER is Undecidable

Machine H

Construct \( M' \) from \( \langle M, w \rangle \)

\( \langle M' \rangle \) (Never Halts)

Decider for DECIDER

What does \( H \) do if \( M \) loops on \( w \)?
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

Simulate $M$ on $w$

(Ignored)
DECIDER is Undecidable

\( \langle M, w \rangle \)

Machine H

Simulate M on w

(Ignored)

Machine M'

x
DECIDER is Undecidable

What does H do if M halts on w?
DECIDER is Undecidable

Machine $H$

$\langle M, w \rangle$

Simulate $M$ on $w$

$Ignored$

What does $H$ do if $M$ halts on $w$?
DECIDER is Undecidable

\[ \langle M, w \rangle \rightarrow \text{Simulate } M \text{ on } w \rightarrow \text{(Ignored)} \]

Machine H

\[ x \rightarrow \text{Simulate } M \text{ on } w \rightarrow \text{Machine } M' \]

(Ignored)
DECIDER is Undecidable

What does H do if M loops on w?
DECIDER is Undecidable

\[ \langle M, w \rangle \]

What does H do if M loops on w?
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

Decider for DECIDER

Machine $H$

Simulate $M$ on $w$

$x$

(Ignored)
DECIDER is Undecidable

Construct $M'$ from $\langle M, w \rangle$

$\langle M' \rangle$

Decider for DECIDER

This is a decider for HALT!
Theorem: $\text{HALT} \leq_{M} \text{DECIDER}$. 
**Theorem:** $HALT \leq^m DECIDER$.

**Proof:** We exhibit a mapping reduction from $HALT$ to $DECIDER$. 

For any TM/string pair $⟨M, w⟩$, let $f(⟨M, w⟩) = ⟨M'⟩$, where $⟨M'⟩$ 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 $⟨M, w⟩ ∈ HALT$ iff $f(⟨M, w⟩) ∈ DECIDER$. 

To see this, note that $f(⟨M, w⟩) = ⟨M'⟩ ∈ DECIDER$ 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 $⟨M, w⟩ ∈ HALT$. Thus $⟨M, w⟩ ∈ HALT$ iff $f(⟨M, w⟩) ∈ DECIDER$. Therefore, $f$ is a mapping reduction from $HALT$ to $DECIDER$, so $HALT \leq^m DECIDER$. ■
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:
**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:

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

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

By the parameterization theorem, \( f \) is a computable function.
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 $f(\langle M, w \rangle) \in \text{DECIDER}$. 
**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. 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:

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

By the parameterization theorem, \( f \) is a computable function. 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:

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

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 \( w \).
Theorem: HALT \leq^M DECIDER.

Proof: We exhibit a mapping reduction from HALT to 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:

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

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

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

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 $\text{HALT} \leq^M \text{DECIDER}$.\[\blacksquare\]
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:

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

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 $\text{HALT} \leq_{m} \text{DECIDER}$. ■
Other Hard Languages

- We can't tell if a TM accepts a specific string.
- Could we determine whether or not a TM accepts one of many different strings with specific properties?
- For example, could we build a TM that determines whether some other TM accepts a string of all 1s?

Let $\text{ONES}_{\text{TM}}$ be the following language:

$$\text{ONES}_{\text{TM}} = \{ \langle M \rangle \mid M \text{ accepts a string of the form } 1^n \}$$

- Is $\text{ONES}_{\text{TM}} \in \mathbb{R}$? Is it RE?
• Unfortunately, $\text{ONES}_{\text{TM}}$ is undecidable.

• However, $\text{ONES}_{\text{TM}}$ is recognizable.
  • Intuition: Nondeterministically guess the string of the form $1^n$ that $M$ will accept, then check that $M$ accepts it.

• We'll show that $\text{ONES}_{\text{TM}}$ is undecidable by showing that $A_{\text{TM}} \leq^M \text{ONES}$. 
\[ A_{TM} \leq_M ONES_{TM} \]

- As before, let's try to find a function \( f \) such that
  \[ \langle M, w \rangle \in A_{TM} \iff f(\langle M, w \rangle) \in ONES_{TM}. \]
- Let's let \( f(\langle M, w \rangle) = \langle M' \rangle \) for some TM \( M' \). Then we want to pick \( M' \) such that
  \[ \langle M, w \rangle \in A_{TM} \iff f(\langle M, w \rangle) \in ONES_{TM} \]
  \[ \langle M, w \rangle \in A_{TM} \iff \langle M' \rangle \in ONES_{TM} \]
  \[ M \text{ accepts } w \iff M' \text{ accepts } 1^n \text{ for some } n \]
The Reduction

- Goal: construct $M'$ so $M'$ accepts $1^n$ for some $n$ iff $M$ accepts $w$.
- Here is one possible option:
  \[
  M' = \text{"On input } x:\]
  Ignore $x$.
  Run $M$ on $w$.
  If $M$ accepts $w$, accept $x$.
  If $M$ rejects $w$, reject $x$.

- As with before, we can justify the construction of $M'$ using the parameterization theorem.
- If $M$ accepts $w$, then $M'$ accepts all strings, including $1^n$ for some $n$.
- If $M$ does not accept $w$, then $M'$ does not accept any strings, so it certainly does not accept any strings of the form $1^n$. 
Theorem: $A_{TM} \leq_M ONES_{TM}$. 

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

On input $x$:
Ignore $x$.
Run $M$ on $w$.
If $M$ accepts $w$, accept $x$.
If $M$ rejects $w$, reject $x$.

By the parameterization theorem, $f$ is a computable function.

We further claim that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in ONES_{TM}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in ONES_{TM}$ iff $M'$ accepts at least one string of the form $1^n$. We claim that $M'$ accepts at least one string of the form $1^n$ iff $M$ accepts $w$. To see this, note that if $M$ accepts $w$, then $M'$ accepts $1^n$, and if $M$ does not accept $w$, then $M'$ rejects all strings, including all strings of the form $1^n$. Finally, $M$ accepts $w$ iff $\langle M, w \rangle \in A_{TM}$. Thus $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in ONES_{TM}$. Consequently, $f$ is a mapping reduction from $A_{TM}$ to $ONES_{TM}$, so $A_{TM} \leq_M ONES_{TM}$ as required. ■
**Theorem:** $A_{TM} \leq_M ONES_{TM}$.

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

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

- If $M$ accepts $w$, accept $x$.
- If $M$ rejects $w$, reject $x$.

By the parameterization theorem, $f$ is a computable function.

We further claim that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in ONES_{TM}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in ONES_{TM}$ iff $M'$ accepts at least one string of the form $1^n$. We claim that $M'$ accepts at least one string of the form $1^n$ iff $M$ accepts $w$. To see this, note that if $M$ accepts $w$, then $M'$ accepts $1$, and if $M$ does not accept $w$, then $M'$ rejects all strings, including all strings of the form $1^n$. Finally, $M$ accepts $w$ iff $\langle M, w \rangle \in A_{TM}$. Thus $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in ONES_{TM}$. Consequently, $f$ is a mapping reduction from $A_{TM}$ to $ONES_{TM}$, so $A_{TM} \leq_M ONES_{TM}$ as required. ■
Theorem: $A_{TM} \leq^m_{M} ONES_{TM}$.

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

$M' = \text{On input } x:
\begin{align*}
&\text{Ignore } x. \\
&\text{Run } M \text{ on } w. \\
&\text{If } M \text{ accepts } w, \text{ accept } x. \\
&\text{If } M \text{ rejects } w, \text{ reject } x.
\end{align*}$

By the parameterization theorem, $f$ is a computable function.

We further claim that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in ONES_{TM}$. To see this, note that $f(\langle M, w \rangle) = \langle M' \rangle \in ONES_{TM}$ iff $M'$ accepts at least one string of the form $1^n$. We claim that $M'$ accepts at least one string of the form $1^n$ iff $M$ accepts $w$. To see this, note that if $M$ accepts $w$, then $M'$ accepts $1$, and if $M$ does not accept $w$, then $M'$ rejects all strings, including all strings of the form $1^n$. Finally, $M$ accepts $w$ iff $\langle M, w \rangle \in A_{TM}$. Thus $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in ONES_{TM}$. Consequently, $f$ is a mapping reduction from $A_{TM}$ to $ONES_{TM}$, so $A_{TM} \leq^m_{M} ONES_{TM}$ as required. ■
Theorem: $A_{TM} \leq^m ONES_{TM}$.

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

$$M' = \text{"On input } x:\text{ Ignore } x.\text{ Run } M \text{ on } w.\text{ If } M \text{ accepts } w, \text{ accept } x.\text{ If } M \text{ rejects } w, \text{ reject } x."$$
Theorem: $A_{\text{TM}} \leq_M \text{ONES}_{\text{TM}}$.

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

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

By the parameterization theorem, $f$ is a computable function.
**Theorem:** $A_{TM} \leq_M ONES_{TM}$.

**Proof:** We exhibit a mapping reduction from $A_{TM}$ to $ONES_{TM}$. For any TM/string pair $\langle M, w \rangle$, let $f(\langle M, w \rangle) = \langle M' \rangle$, where $M'$ 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{ accepts } w, \text{ accept } x.
\text{If } M \text{ rejects } w, \text{ reject } x."

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M, w \rangle \in A_{TM}$ iff $f(\langle M, w \rangle) \in ONES_{TM}$. 

**Theorem:** $A_{\text{TM}} \leq_{M} \text{ONES}_{\text{TM}}$.

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

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

By the parameterization theorem, $f$ is a computable function. We further claim that $(M, w) \in A_{\text{TM}}$ iff $f((M, w)) \in \text{ONES}_{\text{TM}}$. To see this, note that $f((M, w)) = (M') \in \text{ONES}_{\text{TM}}$ iff $M'$ accepts at least one string of the form $1^n$. 


Theorem: \( A_{\text{TM}} \leq_m \text{ONES}_{\text{TM}} \).

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

\[ M' = \text{"On input } x:\]
\[ \text{Ignore } x. \]
\[ \text{Run } M \text{ on } w. \]
\[ \text{If } M \text{ accepts } w, \text{ accept } x. \]
\[ \text{If } M \text{ rejects } w, \text{ reject } x.\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M, w \rangle \in A_{\text{TM}} \) iff \( f(\langle M, w \rangle) \in \text{ONES}_{\text{TM}} \). To see this, note that \( f(\langle M, w \rangle) = \langle M' \rangle \in \text{ONES}_{\text{TM}} \) iff \( M' \) accepts at least one string of the form \( 1^n \). We claim that \( M' \) accepts at least one string of the form \( 1^n \) iff \( M \) accepts \( w \).
**Theorem:** \( A_{TM} \leq_M ONES_{TM} \).

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

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

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M, w \rangle \in A_{TM} \) iff \( f(\langle M, w \rangle) \in ONES_{TM} \). To see this, note that \( f(\langle M, w \rangle) = \langle M' \rangle \in ONES_{TM} \) iff \( M' \) accepts at least one string of the form \( 1^n \). We claim that \( M' \) accepts at least one string of the form \( 1^n \) iff \( M \) accepts \( w \). To see this, note that if \( M \) accepts \( w \), then \( M' \) accepts \( 1 \), and if \( M \) does not accept \( w \), then \( M' \) rejects all strings, including all strings of the form \( 1^n \).
Theorem: $A_{\text{TM}} \leq_M \text{ONES}_{\text{TM}}$.

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

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

By the parameterization theorem, $f$ is a computable function. We further claim that $(M, w) \in A_{\text{TM}}$ iff $f((M, w)) \in \text{ONES}_{\text{TM}}$. To see this, note that $f((M, w)) = (M') \in \text{ONES}_{\text{TM}}$ iff $M'$ accepts at least one string of the form $1^n$. We claim that $M'$ accepts at least one string of the form $1^n$ iff $M$ accepts $w$. To see this, note that if $M$ accepts $w$, then $M'$ accepts $1$, and if $M$ does not accept $w$, then $M'$ rejects all strings, including all strings of the form $1^n$. Finally, $M$ accepts $w$ iff $(M, w) \in A_{\text{TM}}$. 
Theorem: $A_{TM} \leq_M ONES_{TM}$.

Proof: We exhibit a mapping reduction from $A_{TM}$ to $ONES_{TM}$. For any TM/string pair $⟨M, w⟩$, let $f(⟨M, w⟩) = ⟨M'⟩$, where $M'$ is defined in terms of $M$ and $w$ as follows:

$$M' = "On input x:
    Ignore x.
    Run M on w.
    If M accepts w, accept x.
    If M rejects w, reject x."$$

By the parameterization theorem, $f$ is a computable function. We further claim that $⟨M, w⟩ \in A_{TM}$ iff $f(⟨M, w⟩) \in ONES_{TM}$. To see this, note that $f(⟨M, w⟩) = ⟨M'⟩ \in ONES_{TM}$ iff $M'$ accepts at least one string of the form $1^n$. We claim that $M'$ accepts at least one string of the form $1^n$ iff $M$ accepts $w$. To see this, note that if $M$ accepts $w$, then $M'$ accepts $1$, and if $M$ does not accept $w$, then $M'$ rejects all strings, including all strings of the form $1^n$. Finally, $M$ accepts $w$ iff $⟨M, w⟩ \in A_{TM}$. Thus $⟨M, w⟩ \in A_{TM}$ iff $f(⟨M, w⟩) \in ONES_{TM}$.
**Theorem:** \( A_{\text{TM}} \leq M \text{ONES}_{\text{TM}} \).

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

\[
M' = \text{"On input } x:\]
\[
\text{Ignore } x. \\
\text{Run } M \text{ on } w. \\
\text{If } M \text{ accepts } w, \text{ accept } x. \\
\text{If } M \text{ rejects } w, \text{ reject } x."
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M, w \rangle \in A_{\text{TM}} \) iff \( f(\langle M, w \rangle) \in \text{ONES}_{\text{TM}} \). To see this, note that \( f(\langle M, w \rangle) = \langle M' \rangle \in \text{ONES}_{\text{TM}} \) iff \( M' \) accepts at least one string of the form \( 1^n \). We claim that \( M' \) accepts at least one string of the form \( 1^n \) iff \( M \) accepts \( w \). To see this, note that if \( M \) accepts \( w \), then \( M' \) accepts \( 1 \), and if \( M \) does not accept \( w \), then \( M' \) rejects all strings, including all strings of the form \( 1^n \). Finally, \( M \) accepts \( w \) iff \( \langle M, w \rangle \in A_{\text{TM}} \). Thus \( \langle M, w \rangle \in A_{\text{TM}} \) iff \( f(\langle M, w \rangle) \in \text{ONES}_{\text{TM}} \). Consequently, \( f \) is a mapping reduction from \( A_{\text{TM}} \) to \( \text{ONES}_{\text{TM}} \), so \( A_{\text{TM}} \leq M \text{ONES}_{\text{TM}} \) as required.
**Theorem:** $A_{\text{TM}} \leq_M \text{ONES}_{\text{TM}}$.

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

$$M' = \text{“On input } x:\n$$

Ignore $x$.

Run $M$ on $w$.

If $M$ accepts $w$, accept $x$.

If $M$ rejects $w$, reject $x$.”

By the parameterization theorem, $f$ is a computable function. We further claim that $(M, w) \in A_{\text{TM}}$ iff $f((M, w)) \in \text{ONES}_{\text{TM}}$. To see this, note that $f((M, w)) = (M') \in \text{ONES}_{\text{TM}}$ iff $M'$ accepts at least one string of the form $1^n$. We claim that $M'$ accepts at least one string of the form $1^n$ iff $M$ accepts $w$. To see this, note that if $M$ accepts $w$, then $M'$ accepts $1$, and if $M$ does not accept $w$, then $M'$ rejects all strings, including all strings of the form $1^n$. Finally, $M$ accepts $w$ iff $(M, w) \in A_{\text{TM}}$. Thus $(M, w) \in A_{\text{TM}}$ iff $f((M, w)) \in \text{ONES}_{\text{TM}}$. Consequently, $f$ is a mapping reduction from $A_{\text{TM}}$ to $\text{ONES}_{\text{TM}}$, so $A_{\text{TM}} \leq_M \text{ONES}_{\text{TM}}$ as required. ■
A Slightly Modified Question

• We cannot determine whether or not a TM will accept at least one string of all 1s.

• Can we determine whether a TM only accepts strings of all 1s?

• In other words, for a TM $M$, is $\mathcal{L}(M) \subseteq 1^*$?

• Let $\text{ONLYONES}_{TM}$ be the language

\[
\text{ONLYONES}_{TM} = \{ \langle M \rangle \mid \mathcal{L}(M) \subseteq 1^* \}
\]

• Is $\text{ONLYONES}_{TM} \in \text{R}$? How about $\text{RE}$?
ONLYONES$_{\text{TM}} \not\in \text{RE}$

- It turns out that the language ONLYONES$_{\text{TM}}$ is unrecognizable.
- We can prove this by reducing $L_D$ to ONLYONES$_{\text{TM}}$.
- If $L_D \leq_M$ ONLYONES$_{\text{TM}}$, then ONLYONES$_{\text{TM}} \not\in \text{RE}$. 
\[ \mathcal{L}_D \leq_\mathcal{M} \text{ONLYONES}_{\mathcal{TM}} \]

- We want to find a computable function \( f \) such that
  \[ \langle M \rangle \in \mathcal{L}_D \iff f(\langle M \rangle) \in \text{ONLYONES}_{\mathcal{TM}}. \]

- We want to set \( f(\langle M \rangle) = \langle M' \rangle \) for some suitable choice of \( M' \). This means
  \[ \langle M \rangle \in \mathcal{L}_D \iff \langle M' \rangle \in \text{ONLYONES}_{\mathcal{TM}} \]
  \[ \langle M \rangle \notin \mathcal{L}(M) \iff \mathcal{L}(M') \subseteq 1^* \]

- How would we pick our machine \( M' \)?
One Possible Reduction

- We want to build $M'$ from $M$ such that $\langle M \rangle \notin \mathcal{L}(M)$ iff $\mathcal{L}(M') \subseteq 1^*$. 

- In other words, we construct $M'$ such that
  - If $\langle M \rangle \in \mathcal{L}(M)$, then $\mathcal{L}(M')$ is not a subset of $1^*$.
  - If $\langle M \rangle \notin \mathcal{L}(M)$, then $\mathcal{L}(M')$ is a subset of $1^*$.

- One option: Come up with some languages with these properties, then construct our machine $M'$ such that its language changes based on whether $\langle M \rangle \in \mathcal{L}(M)$.
  - If $\langle M \rangle \in \mathcal{L}(M)$, then $\mathcal{L}(M') = \Sigma^*$, which is not a subset of $1^*$.
  - If $\langle M \rangle \notin \mathcal{L}(M)$, then $\mathcal{L}(M') = \emptyset$, which is a subset of $1^*$. 
One Possible Reduction

- We want
  - If \( \langle M \rangle \in \mathcal{L}(M) \), then \( \mathcal{L}(M') = \Sigma^* \)
  - If \( \langle M \rangle \notin \mathcal{L}(M) \), then \( \mathcal{L}(M') = \emptyset \)
- Here is one possible \( M' \) that does this:

  \[ M' = \text{“On input } x:\text{ Ignore } x. \text{ Run } M \text{ on } \langle M \rangle. \text{ If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \text{ If } M \text{ rejects } \langle M \rangle, \text{ reject } x.” \]
Theorem: $L_D \leq_M {\text{ONLYONES}}_{TM}$. 

Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{TM}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

- On input $x$: Ignore $x$. Run $M$ on $\langle M \rangle$. If $M$ accepts $\langle M \rangle$, accept $x$. If $M$ rejects $\langle M \rangle$, reject $x$.

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{TM}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{TM}$ iff $(\mathcal{L}_{M'}) \subseteq 1^*$. We claim that $(\mathcal{L}_{M'}) \subseteq 1^*$ iff $M$ does not accept $\langle M \rangle$. To see this, note that if $M$ does not accept $\langle M \rangle$, then $M'$ never accepts any strings, so $(\mathcal{L}_{M'}) = \emptyset \subseteq 1^*$. Otherwise, if $M$ accepts $\langle M \rangle$, then $M'$ accepts all strings, so $(\mathcal{L}_{M'}) = \Sigma^*$, which is not a subset of $1^*$. Finally, $M$ does not accept $\langle M \rangle$ iff $\langle M \rangle \in L_D$. Thus $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{TM}$. Consequently, $f$ is a mapping reduction from $L_D$ to $\text{ONLYONES}_{TM}$, so $L_D \leq_M \text{ONLYONES}_{TM}$ as required. ■
Theorem: $L_D \leq^M \text{ONLYONES}_{\text{TM}}$.
Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$. 
Theorem: $L_D \leq_M \text{ONLYONES}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:
**Theorem:** $L_D \leq_M \text{ONLYONES}_\text{TM}$.

**Proof:** We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_\text{TM}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\text{ Ignore } x.\text{ Run } M \text{ on } \langle M \rangle.\text{ If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\text{ If } M \text{ rejects } \langle M \rangle, \text{ reject } x."$$
**Theorem:** \( L_D \leq_M \text{ONLYONES}_{TM} \).

**Proof:** We exhibit a mapping reduction from \( L_D \) to \( \text{ONLYONES}_{TM} \).

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

\[
M' = "\text{On input } x:
\quad \text{Ignore } x.
\quad \text{Run } M \text{ on } \langle M \rangle.
\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.
\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, \( f \) is a computable function.
Theorem: $L_D \leq_M \text{ONLYONES}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\text{ Ignore } x. \text{ Run } M \text{ on } \langle M \rangle. \text{ If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \text{ If } M \text{ rejects } \langle M \rangle, \text{ reject } x."$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}}$. 
Theorem: \( L_D \leq_M \text{ONLYONES}_{\text{TM}} \).

Proof: We exhibit a mapping reduction from \( L_D \) to \( \text{ONLYONES}_{\text{TM}} \).

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

\[
M' = \text{"On input } x:\n\text{Ignore } x.\n\text{Run } M \text{ on } \langle M \rangle.\n\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."\n\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}} \). To see this, note that \( f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{\text{TM}} \) iff \( \mathcal{A}(M') \subseteq 1^* \).
Theorem: $L_D \leq_M \text{ONLYONES}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{“On input } x:\text{ Ignore } x.\text{ Run } M \text{ on } \langle M \rangle.\text{ If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\text{ If } M \text{ rejects } \langle M \rangle, \text{ reject } x.”$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{\text{TM}}$ iff $\mathcal{L}(M') \subseteq 1^*$. We claim that $\mathcal{L}(M') \subseteq 1^*$ iff $M$ does not accept $\langle M \rangle$. 
Theorem: $L_D \leq_M \text{ONLYONES}_{TM}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{TM}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\ $$

- Ignore $x$.
- Run $M$ on $\langle M \rangle$.
- If $M$ accepts $\langle M \rangle$, accept $x$.
- If $M$ rejects $\langle M \rangle$, reject $x$.

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{TM}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{TM}$ iff $\mathcal{L}(M') \subseteq 1^*$. We claim that $\mathcal{L}(M') \subseteq 1^*$ iff $M$ does not accept $\langle M \rangle$. To see this, note that if $M$ does not accept $\langle M \rangle$, then $M'$ never accepts any strings, so $\mathcal{L}(M') = \emptyset \subseteq 1^*$.
**Theorem:** $L_D \leq_M \text{ONLYONES}_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$M' = \text{"On input } x:\$

- Ignore $x$.
- Run $M$ on $\langle M \rangle$.
- If $M$ accepts $\langle M \rangle$, accept $x$.
- If $M$ rejects $\langle M \rangle$, reject $x$.

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{\text{TM}}$ iff $\mathcal{L}(M') \subseteq \Sigma^*$. We claim that $\mathcal{L}(M') \subseteq \Sigma^*$ iff $M$ does not accept $\langle M \rangle$. To see this, note that if $M$ does not accept $\langle M \rangle$, then $M'$ never accepts any strings, so $\mathcal{L}(M') = \emptyset \subseteq \Sigma^*$. Otherwise, if $M$ accepts $\langle M \rangle$, then $M'$ accepts all strings, so $\mathcal{L}(M) = \Sigma^*$, which is not a subset of $\Sigma^*$.
**Theorem:** $L_D \leq_M \text{ONLYONES}_{TM}$.

**Proof:** We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{TM}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\text{ Ignore } x. \text{ Run } M \text{ on } \langle M \rangle. \text{ If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \text{ If } M \text{ rejects } \langle M \rangle, \text{ reject } x."$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{TM}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{TM}$ iff $\mathcal{L}(M') \subseteq 1^*$. We claim that $\mathcal{L}(M') \subseteq 1^*$ iff $M$ does not accept $\langle M \rangle$. To see this, note that if $M$ does not accept $\langle M \rangle$, then $M'$ never accepts any strings, so $\mathcal{L}(M') = \emptyset \subseteq 1^*$. Otherwise, if $M$ accepts $\langle M \rangle$, then $M'$ accepts all strings, so $\mathcal{L}(M) = \Sigma^*$, which is not a subset of $1^*$. Finally, $M$ does not accept $\langle M \rangle$ iff $\langle M \rangle \in L_D$. ■
Theorem: $L_D \leq_M \text{ONLYONES}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\"$$

Ignore $x$.
Run $M$ on $\langle M \rangle$.
If $M$ accepts $\langle M \rangle$, accept $x$.
If $M$ rejects $\langle M \rangle$, reject $x$.”

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{\text{TM}}$ iff $\mathcal{A}(M') \subseteq 1^*$. We claim that $\mathcal{A}(M') \subseteq 1^*$ iff $M$ does not accept $\langle M \rangle$. To see this, note that if $M$ does not accept $\langle M \rangle$, then $M'$ never accepts any strings, so $\mathcal{A}(M') = \emptyset \subseteq 1^*$. Otherwise, if $M$ accepts $\langle M \rangle$, then $M'$ accepts all strings, so $\mathcal{A}(M) = \Sigma^*$, which is not a subset of $1^*$. Finally, $M$ does not accept $\langle M \rangle$ iff $\langle M \rangle \in L_D$. Thus $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}}$. 

Theorem: $L_D \leq_M \text{ONLYONES}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = "\text{On input } x:\n\begin{align*}
\text{Ignore } x. \\
\text{Run } M \text{ on } \langle M \rangle. \\
\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \\
\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\end{align*}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{ONLYONES}_{\text{TM}}$ iff $\mathcal{A}(M') \subseteq 1^*$. We claim that $\mathcal{A}(M') \subseteq 1^*$ iff $M$ does not accept $\langle M \rangle$. To see this, note that if $M$ does not accept $\langle M \rangle$, then $M'$ never accepts any strings, so $\mathcal{A}(M') = \emptyset \subseteq 1^*$. Otherwise, if $M$ accepts $\langle M \rangle$, then $M'$ accepts all strings, so $\mathcal{A}(M) = \Sigma^*$, which is not a subset of $1^*$. Finally, $M$ does not accept $\langle M \rangle$ iff $\langle M \rangle \in L_D$. Thus $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{ONLYONES}_{\text{TM}}$. Consequently, $f$ is a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$, so $L_D \leq_M \text{ONLYONES}_{\text{TM}}$ as required.
**Theorem:** $L_D \leq_M \text{ONLYONES}_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$.

For any TM $M$, let $f(⟨M⟩) = ⟨M'⟩$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x: \text{ 
  Ignore } x. 
  \text{ Run } M \text{ on } ⟨M⟩. 
  \text{ If } M \text{ accepts } ⟨M⟩, \text{ accept } x. 
  \text{ If } M \text{ rejects } ⟨M⟩, \text{ reject } x."$$

By the parameterization theorem, $f$ is a computable function.

We further claim that $⟨M⟩ \in L_D$ iff $f(⟨M⟩) \in \text{ONLYONES}_{\text{TM}}$. To see this, note that $f(⟨M⟩) = ⟨M'⟩ \in \text{ONLYONES}_{\text{TM}}$ iff $ℒ(M') \subseteq 1^*$. We claim that $ℒ(M') \subseteq 1^*$ iff $M$ does not accept $⟨M⟩$. To see this, note that if $M$ does not accept $⟨M⟩$, then $M'$ never accepts any strings, so $ℒ(M') = ∅ \subseteq 1^*$. Otherwise, if $M$ accepts $⟨M⟩$, then $M'$ accepts all strings, so $ℒ(M) = Σ^*$, which is not a subset of $1^*$. Finally, $M$ does not accept $⟨M⟩$ iff $⟨M⟩ \in L_D$. Thus $⟨M⟩ \in L_D$ iff $f(⟨M⟩) \in \text{ONLYONES}_{\text{TM}}$. Consequently, $f$ is a mapping reduction from $L_D$ to $\text{ONLYONES}_{\text{TM}}$, so $L_D \leq_M \text{ONLYONES}_{\text{TM}}$ as required. ■
ONLYONES™

• Although ONLYONES™ is not RE, its complement (ONLYONES™) is RE:

  \{ \langle M \rangle \mid \mathcal{L}(M) \text{ is not a subset of } 1^* \} 

• Intuition: Can nondeterministically guess a string in \( \mathcal{L}(M) \) that is not of the form \( 1^n \), then check that \( M \) accepts it.
RE and co-RE

- The class **RE** is the set of languages that are recognized by a TM.
  - Intuitively, problems where a TM can check that an answer is correct.
- The class **co-RE** is the set of languages whose *complements* are recognized by a TM.
  - Intuitively, problems where a TM can check that an answer is *incorrect*.
- A language in co-RE is called **co-recognizable**. A language not in co-RE is called **co-unrecognizable**.
Why **RE** and co-**RE** Matter

- **RE** and co-**RE** are, in a sense, the weakest conditions necessary for a problem to even be attempted by a computer:
  - If a problem is in **RE**, there is a mechanical procedure for verifying correct answers to that problem.
  - If a problem is in co-**RE**, there is a mechanical procedure for refuting incorrect answers to that problem.
- Understanding what problems are in **RE** and co-**RE** will help give a better understanding of what problems can and cannot be solved.
Properties of co-RE

• Recall:
  If $L \in \text{RE}$ and $\overline{L} \in \text{RE}$, then $L \in \mathcal{R}$.

• Rewritten in terms of co-RE:
  If $L \in \text{RE}$ and $L \in \text{co-RE}$, then $L \in \mathcal{R}$.

• Contrapositive:
  If $L \not\in \mathcal{R}$, then $L \not\in \text{RE}$ or $L \not\in \text{co-RE}$ (or both)

• Important results:
  If $L \not\in \mathcal{R}$ and $L \in \text{RE}$, then $L \not\in \text{co-RE}$.
  If $L \not\in \mathcal{R}$ and $L \in \text{co-RE}$, then $L \not\in \text{RE}$. 
The Limits of Computability

- A TM
- DOGWALK
- ADD
- ONLYONES TM
- ONES TM
- HALT
- 0*1*
- R
- RE
- co-RE

All Languages
The Diagonalization Language Revisited

- The diagonalization language $L_D$ is the language
  \[ L_D = \{ \langle M \rangle \mid M \notin \mathcal{L}(M) \} \]

- As we saw before, $L_D \notin \text{RE}$.

- So where is $L_D$? Is it in $L_D \in \text{co-RE}$? Or is it someplace else?
All Turing machines, listed in some order.
All descriptions of TMs, listed in the same order.
<table>
<thead>
<tr>
<th>$\langle M_0 \rangle$</th>
<th>$\langle M_1 \rangle$</th>
<th>$\langle M_2 \rangle$</th>
<th>$\langle M_3 \rangle$</th>
<th>$\langle M_4 \rangle$</th>
<th>$\langle M_5 \rangle$</th>
<th>...</th>
</tr>
</thead>
</table>
| $M_0$             | Acc               | No                | No                | Acc               | Acc               | No  | ...
<p>| $M_1$             |                   |                   |                   |                   |                   |     |
| $M_2$             |                   |                   |                   |                   |                   |     |
| $M_3$             |                   |                   |                   |                   |                   |     |
| $M_4$             |                   |                   |                   |                   |                   |     |
| $M_5$             |                   |                   |                   |                   |                   |     |
| ...               |                   |                   |                   |                   |                   |     |</p>
<table>
<thead>
<tr>
<th>( \langle M_0 \rangle )</th>
<th>( \langle M_1 \rangle )</th>
<th>( \langle M_2 \rangle )</th>
<th>( \langle M_3 \rangle )</th>
<th>( \langle M_4 \rangle )</th>
<th>( \langle M_5 \rangle )</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M_0 )</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>( M_1 )</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>( M_2 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( M_3 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( M_4 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( M_5 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\langle M_0 \rangle)</td>
<td>(\langle M_1 \rangle)</td>
<td>(\langle M_2 \rangle)</td>
<td>(\langle M_3 \rangle)</td>
<td>(\langle M_4 \rangle)</td>
<td>(\langle M_5 \rangle)</td>
<td>...</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>------</td>
</tr>
<tr>
<td><strong>M_0</strong></td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td><strong>M_1</strong></td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td><strong>M_2</strong></td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td><strong>M_3</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>M_4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>M_5</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>...</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\langle M_0 \rangle$</td>
<td>$\langle M_1 \rangle$</td>
<td>$\langle M_2 \rangle$</td>
<td>$\langle M_3 \rangle$</td>
<td>$\langle M_4 \rangle$</td>
<td>$\langle M_5 \rangle$</td>
</tr>
<tr>
<td>---</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>$M_0$</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>$M_1$</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>$M_2$</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>$M_3$</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>$M_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$M_0$</td>
<td>$M_1$</td>
<td>$M_2$</td>
<td>$M_3$</td>
<td>$M_4$</td>
<td>$M_5$</td>
</tr>
<tr>
<td>---</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>$M_0$</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>$M_1$</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>$M_2$</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>$M_3$</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>$M_4$</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>$M_5$</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>(M_0)</td>
<td>(M_1)</td>
<td>(M_2)</td>
<td>(M_3)</td>
<td>(M_4)</td>
<td>(M_5)</td>
<td>...</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-----</td>
</tr>
<tr>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>...</td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>...</td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>...</td>
</tr>
<tr>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>...</td>
</tr>
<tr>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>...</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>(\langle M_0 \rangle)</td>
<td>(\langle M_1 \rangle)</td>
<td>(\langle M_2 \rangle)</td>
<td>(\langle M_3 \rangle)</td>
<td>(\langle M_4 \rangle)</td>
<td>(\langle M_5 \rangle)</td>
</tr>
<tr>
<td>---</td>
<td>------------------</td>
<td>------------------</td>
<td>------------------</td>
<td>------------------</td>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>(M_0)</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>(M_1)</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>(M_2)</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>(M_3)</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>(M_4)</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>(M_5)</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
</tr>
<tr>
<td>\langle M_0 \rangle</td>
<td>\langle M_1 \rangle</td>
<td>\langle M_2 \rangle</td>
<td>\langle M_3 \rangle</td>
<td>\langle M_4 \rangle</td>
<td>\langle M_5 \rangle</td>
<td>...</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td>-----</td>
</tr>
<tr>
<td>M_0</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>M_1</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>M_2</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>M_3</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td>M_4</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td>M_5</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$\langle M_0 \rangle$</td>
<td>$\langle M_1 \rangle$</td>
<td>$\langle M_2 \rangle$</td>
<td>$\langle M_3 \rangle$</td>
<td>$\langle M_4 \rangle$</td>
<td>$\langle M_5 \rangle$</td>
<td>…</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>---------------------</td>
<td>-----------</td>
</tr>
<tr>
<td><strong>M</strong>&lt;sub&gt;0&lt;/sub&gt;</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td><strong>M</strong>&lt;sub&gt;1&lt;/sub&gt;</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td><strong>M</strong>&lt;sub&gt;2&lt;/sub&gt;</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td><strong>M</strong>&lt;sub&gt;3&lt;/sub&gt;</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
</tr>
<tr>
<td><strong>M</strong>&lt;sub&gt;4&lt;/sub&gt;</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
</tr>
<tr>
<td><strong>M</strong>&lt;sub&gt;5&lt;/sub&gt;</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>…</td>
</tr>
</tbody>
</table>
This language is \( L_D \).

<table>
<thead>
<tr>
<th>( M_0 )</th>
<th>( M_1 )</th>
<th>( M_2 )</th>
<th>( M_3 )</th>
<th>( M_4 )</th>
<th>( M_5 )</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>…</td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>…</td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>…</td>
</tr>
<tr>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>…</td>
</tr>
<tr>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>…</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>…</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

\[ \langle M_0 \rangle \langle M_1 \rangle \langle M_2 \rangle \langle M_3 \rangle \langle M_4 \rangle \langle M_5 \rangle \ldots \]

Acc Acc Acc No Acc No …
The Language $\overline{L}_D$

- The language $\overline{L}_D$ is the language
  \[ \overline{L}_D = \{ \langle M \rangle | \langle M \rangle \in \mathcal{L}(M) \} \]
- That is, the set of TMs that accept their own description.
- $L_d \in \text{co-RE}$ iff $\overline{L}_D \in \text{RE}$.
- So is $\overline{L}_D \in \text{RE}$?
\( L_D \in \text{co-RE} \)

- Here's an TM for \( \overline{L}_D \):

  \[ R = \text{"On input } \langle M \rangle \text{:
      Run } M \text{ on } \langle M \rangle.
      \text{If } M \text{ accepts } \langle M \rangle, \text{ accept.}
      \text{If } M \text{ rejects } \langle M \rangle, \text{ reject."
    } \]

- Then \( R \) accepts \( \langle M \rangle \) iff \( \langle M \rangle \in \mathcal{L}(M) \) iff \( \langle M \rangle \in \overline{L}_D \), so \( \mathcal{L}(R) = \overline{L}_D \).
The Limits of Computability

- \( A_{TM} \)
- \( L_D \)
- \( \overline{HALT} \)
- \( \overline{ONES}_{TM} \)
- \( \overline{ONLYONES}_{TM} \)
- \( \text{DOGWALK} \)
- \( \text{ADD} \)
- \( 0^*1^* \)

The diagram shows the relationships between different languages and their complements in the context of computability theory. The areas represent RE (Recursively Enumerable), co-RE (co-Recursively Enumerable), and the intersection represents the RE languages. The specific languages and their properties are highlighted within the diagram.
*Theorem:* If $A \leq_M B$, then $\overline{A} \leq_M \overline{B}$. 
Theorem: If $A \leq_M B$, then $\overline{A} \leq_M \overline{B}$.

Proof: Suppose that $A \leq_M B$. 
**Theorem**: If $A \leq_M B$, then $\overline{A} \leq_M \overline{B}$.

**Proof**: Suppose that $A \leq_M B$. Then there exists a computable function $f$ such that $w \in A$ iff $f(w) \in B$. Note that $w \in A$ iff $w \not\in A$ and $f(w) \in B$ iff $f(w) \not\in B$. Consequently, we have that $w \not\in A$ iff $f(w) \not\in B$. Thus $w \in A$ iff $f(w) \in B$. Since $f$ is computable, $A \leq_M B$. ■
**Theorem:** If $A \leq_M B$, then $\overline{A} \leq_M \overline{B}$.

**Proof:** Suppose that $A \leq_M B$. Then there exists a computable function $f$ such that $w \in A$ iff $f(w) \in B$. Note that $w \in A$ iff $w \notin \overline{A}$ and $f(w) \in B$ iff $f(w) \notin \overline{B}$.
**Theorem:** If $A \leq M B$, then $\overline{A} \leq M \overline{B}$.

**Proof:** Suppose that $A \leq M B$. Then there exists a computable function $f$ such that $w \in A$ iff $f(w) \in B$. Note that $w \in A$ iff $w \notin \overline{A}$ and $f(w) \in B$ iff $f(w) \notin \overline{B}$. Consequently, we have that $w \notin \overline{A}$ iff $f(w) \notin \overline{B}$. 

Theorem: If $A \leq_M B$, then $\overline{A} \leq_M \overline{B}$.

Proof: Suppose that $A \leq_M B$. Then there exists a computable function $f$ such that $w \in A$ iff $f(w) \in B$. Note that $w \in A$ iff $w \notin \overline{A}$ and $f(w) \in B$ iff $f(w) \notin \overline{B}$. Consequently, we have that $w \notin \overline{A}$ iff $f(w) \notin \overline{B}$. Thus $w \in \overline{A}$ iff $f(w) \in \overline{B}$. ■
Theorem: If $A \leq_M B$, then $\overline{A} \leq_M \overline{B}$.

Proof: Suppose that $A \leq_M B$. Then there exists a computable function $f$ such that $w \in A$ iff $f(w) \in B$. Note that $w \in A$ iff $w \notin \overline{A}$ and $f(w) \in B$ iff $f(w) \notin \overline{B}$. Consequently, we have that $w \notin \overline{A}$ iff $f(w) \notin \overline{B}$. Thus $w \in \overline{A}$ iff $f(w) \in \overline{B}$. Since $f$ is computable, $\overline{A} \leq_M \overline{B}$. ■
Theorem: If $A \leq_M B$, then $\overline{A} \leq_M \overline{B}$.

Proof: Suppose that $A \leq_M B$. Then there exists a computable function $f$ such that $w \in A$ iff $f(w) \in B$. Note that $w \in A$ iff $w \notin \overline{A}$ and $f(w) \in B$ iff $f(w) \notin \overline{B}$. Consequently, we have that $w \notin \overline{A}$ iff $f(w) \notin \overline{B}$. Thus $w \in \overline{A}$ iff $f(w) \in \overline{B}$. Since $f$ is computable, $\overline{A} \leq_M \overline{B}$. ■
co-RE Reductions

- **Corollary**: If $A \leq^M B$ and $B \in \text{co-RE}$, then $A \in \text{co-RE}$.

  *Proof*: Since $A \leq^M B$, $\overline{A} \leq^M \overline{B}$. Since $B \in \text{co-RE}$, $\overline{B} \in \text{RE}$. Thus $\overline{A} \in \text{RE}$, so $A \in \text{co-RE}$. ■

- **Corollary**: If $A \leq^M B$ and $A \not\in \text{co-RE}$, then $B \not\in \text{co-RE}$.

  *Proof*: Take the contrapositive of the above. ■
Why Mapping Reducibility Matters

\[ A \leq M_{\text{M}} B \]

If this one is "easy" (R or RE or co-RE)...

... then this one is "easy" (R or RE or co-RE) too.
Why Mapping Reducibility Matters

If this one is "hard" (not $R$ or not $RE$ or not $co-RE$)...

\[ A \leq_{M} B \]

... then this one is "hard" (not $R$ or not $RE$ or not $co-RE$) too.
Is there anything out here?
RE \cup \text{co-RE} \text{ is Not Everything}

- Using the same reasoning as the first day of lecture, we can show that there must be problems that are neither RE nor co-RE.
- There are more sets of strings than TMs.
- There are more sets of strings than twice the number of TMs.
- What do these languages look like?
An Extremely Hard Problem

• Recall: All regular languages are also RE.
• This means that some TMs accept regular languages and some TMs do not.
• Let $\text{REGULAR}_{TM}$ be the language of all TM descriptions that accept regular languages:
  $\text{REGULAR}_{TM} = \{ \langle M \rangle \mid \mathcal{L}(M) \text{ is regular.} \}$
• Is $\text{REGULAR}_{TM} \in \mathbb{R}$? How about RE?
REGULAR_{\text{TM}} \notin \text{RE}

- It turns out that REGULAR_{\text{TM}} is unrecognizable, meaning that there is no computer program that can even verify that another TM's language is regular!
- To do this, we'll do another reduction from $L_D$ and prove that $L_D \leq_M \text{REGULAR}_{\text{TM}}$. 
\[ L_D \leq_M \text{REGULAR}_{\text{TM}} \]

- We want to find a computable function \( f \) such that
  \[ \langle M \rangle \in \mathcal{L}_D \iff f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}. \]
- We need to choose \( M' \) such that \( f(\langle M \rangle) = \langle M' \rangle \) for some TM \( M' \). Then
  \[ \langle M \rangle \in \mathcal{L}_D \iff f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}} \]
  \[ \langle M \rangle \in \mathcal{L}_D \iff \langle M' \rangle \in \text{REGULAR}_{\text{TM}} \]
  \[ \langle M \rangle \not\in \mathcal{L}(M) \iff \mathcal{L}(M') \text{ is regular.} \]
\[ L_D \leq_M \text{REGULAR}_{\text{TM}} \]

- We want to construct some \( M' \) out of \( M \) such that
  - If \( \langle M \rangle \in \mathcal{L}(M) \), then \( \mathcal{L}(M') \) is not regular.
  - If \( \langle M \rangle \not\in \mathcal{L}(M) \), then \( \mathcal{L}(M') \) is regular.

- One option: choose two languages, one regular and one nonregular, then construct \( M' \) so its language switches from regular to nonregular based on whether \( \langle M \rangle \not\in \mathcal{L}(M) \).
  - If \( \langle M \rangle \in \mathcal{L}(M) \), then \( \mathcal{L}(M') = \{ 0^n1^n \mid n \in \mathbb{N} \} \)
  - If \( \langle M \rangle \not\in \mathcal{L}(M) \), then \( \mathcal{L}(M') = \emptyset \)
The Reduction

- We want to build $M'$ from $M$ such that
  - If $\langle M \rangle \in \mathcal{L}(M)$, then $\mathcal{L}(M') = \{ 0^n1^n | n \in \mathbb{N} \}$
  - If $\langle M \rangle \notin \mathcal{L}(M)$, then $\mathcal{L}(M') = \emptyset$

- Here is one way to do this:

  $M' = "$On input $x$:
  - If $x$ does not have the form $0^n1^n$, reject.
  - Run $M$ on $\langle M \rangle$.
  - If $M$ accepts, accept $x$.
  - If $M$ rejects, reject $x$."

Theorem: $L_D \leq_M \text{REGULAR}_{TM}$. 

Proof:
We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{TM}$.
For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

- $M' = "\text{On input } x:\n  \text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x.\n  \text{Run } M \text{ on } \langle M \rangle.\n  \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n  \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."."

By the parameterization theorem, $f$ is a computable function.
We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{TM}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{TM}$ iff $(\ell M')$ is regular. We claim that $(\ell M')$ is regular iff $\langle M \rangle \not\in (\ell M)$. To see this, note that if $\langle M \rangle \not\in (\ell M)$, then $M'$ never accepts any strings. Thus $(\ell M') = \emptyset$, which is regular. Otherwise, if $\langle M \rangle \in (\ell M)$, then $M'$ accepts all strings of the form $0^n1^n$, so we have that $(\ell M') = \{0^n1^n | n \in \mathbb{N}\}$, which is not regular. Finally, $\langle M \rangle \not\in (\ell M')$ iff $\langle M \rangle \in L_D$. Thus $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{TM}$, so $f$ is a mapping reduction from $L_D$ to $\text{REGULAR}_{TM}$. Therefore, $L_D \leq_M \text{REGULAR}_{TM}$. ■
Theorem: $L_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{TM}}$. 

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$M'$ = "On input $x$:
1. If $x$ does not have the form $0^n1^n$, reject $x$.
2. Run $M$ on $\langle M \rangle$.
3. If $M$ accepts $\langle M \rangle$, accept $x$.
4. If $M$ rejects $\langle M \rangle$, reject $x".

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ if and only if $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ if and only if $(\mathcal{L}_{M'})$ is regular. We claim that $(\mathcal{L}_{M'})$ is regular if and only if $\langle M \rangle \not\in (\mathcal{L}_{M})$. To see this, note that if $\langle M \rangle \not\in (\mathcal{L}_{M})$, then $M'$ never accepts any strings. Thus $(\mathcal{L}_{M'}) = \emptyset$, which is regular. Otherwise, if $\langle M \rangle \in (\mathcal{L}_{M})$, then $M'$ accepts all strings of the form $0^n1^n$, so we have that $(\mathcal{L}_{M}) = \{0^n1^n | n \in \mathbb{N}\}$, which is not regular. Finally, $\langle M \rangle \not\in (\mathcal{L}_{M'})$ if and only if $\langle M \rangle \in L_D$. Thus $\langle M \rangle \in L_D$ if and only if $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$, so $f$ is a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{TM}}$. Therefore, $L_D \leq_M \text{REGULAR}_{\text{TM}}$. ■
Theorem: $L_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{TM}}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:
Theorem: $L_D \leq^m \text{REGULAR}_\text{TM}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_\text{TM}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = "\text{On input } x:\"
\text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x.
\text{Run } M \text{ on } \langle M \rangle.
\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.
\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
$$
**Theorem:** $L_D \leq_M \text{REGULAR}_{TM}$.

**Proof:** We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{TM}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input $x$:}$$

- If $x$ does not have the form $0^n1^n$, reject $x$.
- Run $M$ on $\langle M \rangle$.
- If $M$ accepts $\langle M \rangle$, accept $x$.
- If $M$ rejects $\langle M \rangle$, reject $x$.

By the parameterization theorem, $f$ is a computable function.
Theorem: $L_D \leq_M \text{REGULAR}_{\text{TM}}$.  

Proof: We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{TM}}$.  

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

\[
M' = "\text{On input } x:\n\quad \text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x.\n\quad \text{Run } M \text{ on } \langle M \rangle.\n\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, $f$ is a computable function.  
We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$.  

Theorem: $L_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{TM}}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$M' = "\text{On input } x: $
- If $x$ does not have the form $0^n1^n$, reject $x$.
- Run $M$ on $\langle M \rangle$.
- If $M$ accepts $\langle M \rangle$, accept $x$.
- If $M$ rejects $\langle M \rangle$, reject $x."$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $A(M')$ is regular.
Theorem: \( L_D \leq_M \text{REGULAR}_{\text{TM}} \).

Proof: We exhibit a mapping reduction from \( L_D \) to \( \text{REGULAR}_{\text{TM}} \).

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

\[
M' = \text{"On input } x:\ \\
\text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x. \\
\text{Run } M \text{ on } \langle M \rangle. \\
\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \\
\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}} \). To see this, note that \( f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}} \) iff \( \mathcal{A}(M') \) is regular. We claim that \( \mathcal{A}(M') \) is regular iff \( \langle M \rangle \notin \mathcal{A}(M) \).
**Theorem:** $L_D \leq_M \text{REGULAR}_{TM}$.

**Proof:** We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{TM}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = "\text{On input } x:\n\quad \text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x.\n\quad \text{Run } M \text{ on } \langle M \rangle.\n\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{TM}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{TM}$ iff $\mathcal{A}(M')$ is regular. We claim that $\mathcal{A}(M')$ is regular iff $\langle M \rangle \notin \mathcal{A}(M)$. To see this, note that if $\langle M \rangle \notin \mathcal{A}(M)$, then $M'$ never accepts any strings.
Theorem: $L_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{TM}}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\n$$
If $x$ does not have the form $0^n1^n$, reject $x$.
Run $M$ on $\langle M \rangle$.
If $M$ accepts $\langle M \rangle$, accept $x$.
If $M$ rejects $\langle M \rangle$, reject $x.$"

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $\mathcal{A}(M')$ is regular. We claim that $\mathcal{A}(M')$ is regular iff $\langle M \rangle \notin \mathcal{A}(M)$. To see this, note that if $\langle M \rangle \notin \mathcal{A}(M)$, then $M'$ never accepts any strings. Thus $\mathcal{A}(M') = \emptyset$, which is regular.
Theorem: \( L_D \leq_M \text{REGULAR}_{\text{TM}} \).

Proof: We exhibit a mapping reduction from \( L_D \) to \( \text{REGULAR}_{\text{TM}} \).

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

\[
M' = "\text{On input } x:\n\quad \text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x.\n\quad \text{Run } M \text{ on } \langle M \rangle.\n\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}} \). To see this, note that \( f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}} \) iff \( \mathcal{A}(M') \) is regular. We claim that \( \mathcal{A}(M') \) is regular iff \( \langle M \rangle \notin \mathcal{A}(M) \). To see this, note that if \( \langle M \rangle \notin \mathcal{A}(M) \), then \( M' \) never accepts any strings. Thus \( \mathcal{A}(M') = \emptyset \), which is regular. Otherwise, if \( \langle M \rangle \in \mathcal{A}(M) \), then \( M' \) accepts all strings of the form \( 0^n1^n \), so we have that \( \mathcal{A}(M) = \{ 0^n1^n \mid n \in \mathbb{N} \} \), which is not regular.
**Theorem:** \( L_D \leq_M \text{REGULAR}_{TM} \).

**Proof:** We exhibit a mapping reduction from \( L_D \) to \( \text{REGULAR}_{TM} \).

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

\[
M' = "On input x:
   If x does not have the form 0^n1^n, reject x.
   Run M on \langle M \rangle.
   If M accepts \langle M \rangle, accept x.
   If M rejects \langle M \rangle, reject x."
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{TM} \). To see this, note that \( f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{TM} \) iff \( \mathcal{A}(M') \) is regular. We claim that \( \mathcal{A}(M') \) is regular iff \( \langle M \rangle \notin \mathcal{A}(M) \). To see this, note that if \( \langle M \rangle \notin \mathcal{A}(M) \), then \( M' \) never accepts any strings. Thus \( \mathcal{A}(M') = \emptyset \), which is regular. Otherwise, if \( \langle M \rangle \in \mathcal{A}(M) \), then \( M' \) accepts all strings of the form \( 0^n1^n \), so we have that \( \mathcal{A}(M) = \{ 0^n1^n \mid n \in \mathbb{N} \} \), which is not regular. Finally, \( \langle M \rangle \notin \mathcal{A}(M') \) iff \( \langle M \rangle \in L_D \).
**Theorem:** \( L_D \leq_M \text{REGULAR}_{TM} \).

**Proof:** We exhibit a mapping reduction from \( L_D \) to \( \text{REGULAR}_{TM} \).

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

\[
M' = "\text{On input } x:\n\begin{align*}
\text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x. \\
\text{Run } M \text{ on } \langle M \rangle. \\
\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \\
\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{TM} \). To see this, note that \( f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{TM} \) iff \( \mathcal{L}(M') \) is regular. We claim that \( \mathcal{L}(M') \) is regular iff \( \langle M \rangle \notin \mathcal{L}(M) \). To see this, note that if \( \langle M \rangle \notin \mathcal{L}(M) \), then \( M' \) never accepts any strings. Thus \( \mathcal{L}(M') = \emptyset \), which is regular. Otherwise, if \( \langle M \rangle \in \mathcal{L}(M) \), then \( M' \) accepts all strings of the form \( 0^n1^n \), so we have that \( \mathcal{L}(M) = \{ 0^n1^n \mid n \in \mathbb{N} \} \), which is not regular. Finally, \( \langle M \rangle \notin \mathcal{L}(M) \) iff \( \langle M \rangle \in L_D \). Thus \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{TM} \), so \( f \) is a mapping reduction from \( L_D \) to \( \text{REGULAR}_{TM} \).
**Theorem:** \( L_D \leq_M \text{REGULAR}_{\text{TM}} \).

**Proof:** We exhibit a mapping reduction from \( L_D \) to \( \text{REGULAR}_{\text{TM}} \).

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

\[
M' = \"On input \( x \):
\begin{align*}
&\text{If } x \text{ does not have the form } 0^n1^n, \text{ reject } x. \\
&\text{Run } M \text{ on } \langle M \rangle. \\
&\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \\
&\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x.\end{align*}
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}} \). To see this, note that \( f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}} \) iff \( \mathcal{A}(M') \) is regular. We claim that \( \mathcal{A}(M') \) is regular iff \( \langle M \rangle \notin \mathcal{A}(M) \). To see this, note that if \( \langle M \rangle \notin \mathcal{A}(M) \), then \( M' \) never accepts any strings. Thus \( \mathcal{A}(M') = \emptyset \), which is regular. Otherwise, if \( \langle M \rangle \in \mathcal{A}(M) \), then \( M' \) accepts all strings of the form \( 0^n1^n \), so we have that \( \mathcal{A}(M) = \{ 0^n1^n \mid n \in \mathbb{N} \} \), which is not regular. Finally, \( \langle M \rangle \notin \mathcal{A}(\langle M \rangle) \) iff \( \langle M \rangle \in L_D \). Thus \( \langle M \rangle \in L_D \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}} \), so \( f \) is a mapping reduction from \( L_D \) to \( \text{REGULAR}_{\text{TM}} \). Therefore, \( L_D \leq_M \text{REGULAR}_{\text{TM}} \).
Theorem: $L_D \leq_M \text{REGULAR}_{\text{Tm}}$.

Proof: We exhibit a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{Tm}}$.

For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{“On input } x:\text{ if } x \text{ does not have the form } 0^n1^n, \text{ reject } x.\text{ Run } M \text{ on } \langle M \rangle.\text{ If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\text{ If } M \text{ rejects } \langle M \rangle, \text{ reject } x.”$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{Tm}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{Tm}}$ iff $\mathcal{L}(M')$ is regular. We claim that $\mathcal{L}(M')$ is regular iff $\langle M \rangle \notin \mathcal{L}(M)$. To see this, note that if $\langle M \rangle \notin \mathcal{L}(M)$, then $M'$ never accepts any strings. Thus $\mathcal{L}(M') = \emptyset$, which is regular. Otherwise, if $\langle M \rangle \in \mathcal{L}(M)$, then $M'$ accepts all strings of the form $0^n1^n$, so we have that $\mathcal{L}(M) = \{ 0^n1^n | n \in \mathbb{N} \}$, which is not regular. Finally, $\langle M \rangle \notin \mathcal{L}(\langle M \rangle)$ iff $\langle M \rangle \in L_D$. Thus $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{Tm}}$, so $f$ is a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{Tm}}$. Therefore, $L_D \leq_M \text{REGULAR}_{\text{Tm}}$. ■
Not only is $\text{REGULAR}_{\text{TM}} \not\in \text{RE}$, but $\text{REGULAR}_{\text{TM}} \not\in \text{co-RE}$.

Before proving this, take a minute to think about just how ridiculously hard this problem is.

- No computer can confirm that an arbitrary TM has a regular language.
- No computer can confirm that an arbitrary TM has a nonregular language.
- This is vastly beyond the limits of what computers could ever hope to solve.
\[ \overline{L}_D \leq_M \text{REGULAR}_{\text{TM}} \]

- To prove that \( \text{REGULAR}_{\text{TM}} \) is not co-\text{RE}, we will prove that \( \overline{L}_D \leq_M \text{REGULAR}_{\text{TM}} \).

- Since \( \overline{L}_D \) is not co-\text{RE}, this proves that \( \text{REGULAR}_{\text{TM}} \) is not co-\text{RE} either.

- Goal: Find a function \( f \) such that

\[
\langle M \rangle \in L_D \quad \text{iff} \quad f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}
\]

- Let \( f(\langle M \rangle) = \langle M' \rangle \) for some TM \( M' \). Then we want

\[
\langle M \rangle \in L_D \quad \text{iff} \quad \langle M' \rangle \in \text{REGULAR}_{\text{TM}}
\]

\[
\langle M \rangle \in \mathcal{L}(M) \quad \text{iff} \quad \mathcal{L}(M') \text{ is regular}
\]
\( \bar{L}_D \leq_M \text{REGULAR}_{TM} \)

- We want to construct some \( M' \) out of \( M \) such that
  - If \( \langle M \rangle \in \mathcal{L}(M) \), then \( \mathcal{L}(M') \) is regular.
  - If \( \langle M \rangle \notin \mathcal{L}(M) \), then \( \mathcal{L}(M') \) is not regular.
- One option: choose two languages, one regular and one nonregular, then construct \( M' \) so its language switches from regular to nonregular based on whether \( \langle M \rangle \in \mathcal{L}(M) \).
\[ \overline{L_D} \leq_m \text{REGULAR}_{TM} \]

- We want to build \( M' \) from \( M \) such that
  - If \( \langle M \rangle \in \mathcal{L}(M) \), then \( \mathcal{L}(M') = \Sigma^* \)
  - If \( \langle M \rangle \notin \mathcal{L}(M) \), then \( \mathcal{L}(M') = \{ \ 0^n1^n \mid n \in \mathbb{N} \ \} \)
- Here is one way to do this:

\[ M' = \text{"On input } x:\]
\[ \quad \text{If } x \text{ has the form } 0^n1^n, \text{ accept.} \]
\[ \quad \text{Run } M \text{ on } \langle M \rangle. \]
\[ \quad \text{If } M \text{ accepts, accept } x. \]
\[ \quad \text{If } M \text{ rejects, reject } x. \]
Theorem: $\overline{L}_D \leq_{M} \text{REGULAR}_{TM}$.
Theorem: $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. 

For any $\text{TM} M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

- **On input $x$:**
  - If $x$ has the form $0^n1^n$, accept $x$.
  - Run $M$ on $\langle M \rangle$.
  - If $M$ accepts $\langle M \rangle$, accept $x$.
  - If $M$ rejects $\langle M \rangle$, reject $x$.

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $(\mathcal{L}_{M'})$ is regular. We claim that $\mathcal{L}(M')$ is regular iff $\langle M \rangle \in (\mathcal{L}_M)$. To see this, note that if $\langle M \rangle \in (\mathcal{L}_M)$, then $M'$ accepts all strings, either because that string is of the form $0^n1^n$ or because $M$ eventually accepts $\langle M \rangle$. Thus $(\mathcal{L}_{M'}) = \Sigma^*$, which is regular. Otherwise, if $\langle M \rangle \notin (\mathcal{L}_M)$, then $M'$ only accepts strings of the form $0^n1^n$, so $(\mathcal{L}_{M'}) = \{0^n1^n | n \in \mathbb{N}\}$, which is not regular. Finally, $\langle M \rangle \in (\mathcal{L}_M)$ iff $\langle M \rangle \in L_D$. Thus $\langle M \rangle \in L_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$, so $f$ is a mapping reduction from $L_D$ to $\text{REGULAR}_{\text{TM}}$. Therefore, $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$. ■
Theorem: \( \bar{L}_D \leq_M \text{REGULAR} \)\(_{\text{TM}}\).

Proof: We exhibit a mapping reduction from \( \bar{L}_D \) to \( \text{REGULAR} \)\(_{\text{TM}}\). For any TM \( M \), let \( f(\langle M \rangle) = \langle M' \rangle \), where \( M' \) is defined in terms of \( M \) as follows:
**Theorem:** \( \overline{L}_D \leq_M \text{REGULAR}_{\text{TM}} \).

**Proof:** We exhibit a mapping reduction from \( \overline{L}_D \) to \( \text{REGULAR}_{\text{TM}} \). For any TM \( M \), let \( f(\langle M \rangle) = \langle M' \rangle \), where \( M' \) is defined in terms of \( M \) as follows:

\[
M' = \text{"On input } x: \\
\quad \text{If } x \text{ has the form } 0^n1^n, \text{ accept } x. \\
\quad \text{Run } M \text{ on } \langle M \rangle. \\
\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \\
\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]
Theorem: $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\n\quad \text{If } x \text{ has the form } 0^n1^n, \text{ accept } x.\n\quad \text{Run } M \text{ on } \langle M \rangle.\n\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x.\n$$

By the parameterization theorem, $f$ is a computable function.
**Theorem:** \( \overline{L_D} \leq_M \text{REGULAR}_{TM} \).

**Proof:** We exhibit a mapping reduction from \( \overline{L_D} \) to \( \text{REGULAR}_{TM} \). For any TM \( M \), let \( f(\langle M \rangle) = \langle M' \rangle \), where \( M' \) is defined in terms of \( M \) as follows:

\[
M' = \text{"On input } x: \\
\text{If } x \text{ has the form } 0^n1^n, \text{ accept } x. \\
\text{Run } M \text{ on } \langle M \rangle. \\
\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \\
\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in \overline{L_D} \) iff \( f(\langle M \rangle) \in \text{REGULAR}_{TM} \).
Theorem: $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

\[
M' = \text{"On input } x:\n\]
\[
\text{If } x \text{ has the form } 0^n1^n, \text{ accept } x.
\]
\[
\text{Run } M \text{ on } \langle M \rangle.
\]
\[
\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.
\]
\[
\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x.
\]

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $A(M')$ is regular.
Theorem: $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{On input } x:\n\quad \text{If } x \text{ has the form } 0^n1^n, \text{ accept } x.\n\quad \text{Run } M \text{ on } \langle M \rangle.\n\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x.$$ 

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $A(M')$ is regular. We claim that $A(M')$ is regular iff $\langle M \rangle \in A(M)$. 

Theorem: $\overline{L}_D \leq_M \text{REGULAR}_{TM}$.

Proof: We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{TM}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$M' = "$On input $x$:
  If $x$ has the form $0^n1^n$, accept $x$.
  Run $M$ on $\langle M \rangle$.
  If $M$ accepts $\langle M \rangle$, accept $x$.
  If $M$ rejects $\langle M \rangle$, reject $x$."

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{TM}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{TM}$ iff $\mathcal{A}(M')$ is regular. We claim that $\mathcal{A}(M')$ is regular iff $\langle M \rangle \in \mathcal{A}(M)$. To see this, note that if $\langle M \rangle \in \mathcal{A}(M)$, then $M'$ accepts all strings, either because that string is of the form $0^n1^n$ or because $M$ eventually accepts $\langle M \rangle$. 

**Theorem:** $\overline{L_D} \leq_M \text{REGULAR}_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction from $\overline{L_D}$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x: \quad \text{If } x \text{ has the form } 0^n1^n, \text{ accept } x. \quad \text{Run } M \text{ on } \langle M \rangle. \quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L_D}$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $\mathcal{A}(M')$ is regular. We claim that $\mathcal{A}(M')$ is regular iff $\langle M \rangle \in \mathcal{A}(M)$. To see this, note that if $\langle M \rangle \in \mathcal{A}(M)$, then $M'$ accepts all strings, either because that string is of the form $0^n1^n$ or because $M$ eventually accepts $\langle M \rangle$. Thus $\mathcal{A}(M') = \Sigma^*$, which is regular.
**Theorem:** $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\"$$

- If $x$ has the form $0^n1^n$, accept $x$.
- Run $M$ on $\langle M \rangle$.
- If $M$ accepts $\langle M \rangle$, accept $x$.
- If $M$ rejects $\langle M \rangle$, reject $x$.”

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $\mathcal{A}(M')$ is regular. We claim that $\mathcal{A}(M')$ is regular iff $\langle M \rangle \in \mathcal{A}(M)$. To see this, note that if $\langle M \rangle \in \mathcal{A}(M)$, then $M'$ accepts all strings, either because that string is of the form $0^n1^n$ or because $M$ eventually accepts $\langle M \rangle$. Thus $\mathcal{A}(M') = \Sigma^*$, which is regular. Otherwise, if $\langle M \rangle \notin \mathcal{A}(M)$, then $M'$ only accepts strings of the form $0^n1^n$, so $\mathcal{A}(M) = \{ 0^n1^n \mid n \in \mathbb{N} \}$, which is not regular.
**Theorem:** $\overline{L_D} \leq_M \text{REGULAR}_{\text{TM}}$.

**Proof:** We exhibit a mapping reduction from $\overline{L_D}$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\ \begin{align*}
\text{If } x \text{ has the form } 0^n1^n, \text{ accept } x. \\
\text{Run } M \text{ on } \langle M \rangle. \\
\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \\
\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x. 
\end{align*}$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L_D}$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $\mathcal{A}(M')$ is regular. We claim that $\mathcal{A}(M')$ is regular iff $\langle M \rangle \in \mathcal{A}(M)$. To see this, note that if $\langle M \rangle \in \mathcal{A}(M)$, then $M'$ accepts all strings, either because that string is of the form $0^n1^n$ or because $M$ eventually accepts $\langle M \rangle$. Thus $\mathcal{A}(M') = \Sigma^*$, which is regular. Otherwise, if $\langle M \rangle \notin \mathcal{A}(M)$, then $M'$ only accepts strings of the form $0^n1^n$, so $\mathcal{A}(M) = \{ 0^n1^n | n \in \mathbb{N} \}$, which is not regular. Finally, $\langle M \rangle \in \mathcal{A}(\langle M \rangle)$ iff $\langle M \rangle \in \overline{L_D}$.
Theorem: $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

\[
M' = "\text{On input } x:\n\text{If } x \text{ has the form } 0^n1^n, \text{ accept } x.\\\text{Run } M \text{ on } \langle M \rangle.\\\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\\\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $\mathcal{A}(M')$ is regular. We claim that $\mathcal{A}(M')$ is regular iff $\langle M \rangle \in \mathcal{A}(M)$. To see this, note that if $\langle M \rangle \in \mathcal{A}(M)$, then $M'$ accepts all strings, either because that string is of the form $0^n1^n$ or because $M$ eventually accepts $\langle M \rangle$. Thus $\mathcal{A}(M') = \Sigma^*$, which is regular. Otherwise, if $\langle M \rangle \notin \mathcal{A}(M)$, then $M'$ only accepts strings of the form $0^n1^n$, so $\mathcal{A}(M) = \{0^n1^n \mid n \in \mathbb{N}\}$, which is not regular. Finally, $\langle M \rangle \in \mathcal{A}(\langle M \rangle)$ iff $\langle M \rangle \in \overline{L}_D$. Thus $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$, so $f$ is a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. ■
**Theorem:** \( \overline{L_D} \leq_M \text{REGULAR}_{\text{TM}}. \)

**Proof:** We exhibit a mapping reduction from \( \overline{L_D} \) to \( \text{REGULAR}_{\text{TM}}. \) For any TM \( M \), let \( f(\langle M \rangle) = \langle M' \rangle \), where \( M' \) is defined in terms of \( M \) as follows:

\[
M' = "\text{On input } x:\n\text{If } x \text{ has the form } 0^n1^n, \text{ accept } x.\n\text{Run } M \text{ on } \langle M \rangle.\n\text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x.\n\text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."
\]

By the parameterization theorem, \( f \) is a computable function. We further claim that \( \langle M \rangle \in \overline{L_D} \iff f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}. \) To see this, note that \( f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}} \iff \mathcal{A}(M') \) is regular. We claim that \( \mathcal{A}(M') \) is regular iff \( \langle M \rangle \in \mathcal{A}(M). \) To see this, note that if \( \langle M \rangle \in \mathcal{A}(M), \) then \( M' \) accepts all strings, either because that string is of the form \( 0^n1^n \) or because \( M \) eventually accepts \( \langle M \rangle. \) Thus \( \mathcal{A}(M') = \Sigma^* \), which is regular. Otherwise, if \( \langle M \rangle \notin \mathcal{A}(M), \) then \( M' \) only accepts strings of the form \( 0^n1^n, \) so \( \mathcal{A}(M) = \{ 0^n1^n \mid n \in \mathbb{N} \} \), which is not regular. Finally, \( \langle M \rangle \in \mathcal{A}(\langle M \rangle) \iff \langle M \rangle \in \overline{L_D}. \) Thus \( \langle M \rangle \in \overline{L_D} \iff f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}, \) so \( f \) is a mapping reduction from \( \overline{L_D} \) to \( \text{REGULAR}_{\text{TM}}. \) Therefore, \( \overline{L_D} \leq_M \text{REGULAR}_{\text{TM}}. \)
Theorem: $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$.

Proof: We exhibit a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. For any TM $M$, let $f(\langle M \rangle) = \langle M' \rangle$, where $M'$ is defined in terms of $M$ as follows:

$$M' = \text{"On input } x:\n\quad \text{If } x \text{ has the form } 0^n1^n, \text{ accept } x. \n\quad \text{Run } M \text{ on } \langle M \rangle. \n\quad \text{If } M \text{ accepts } \langle M \rangle, \text{ accept } x. \n\quad \text{If } M \text{ rejects } \langle M \rangle, \text{ reject } x."$$

By the parameterization theorem, $f$ is a computable function. We further claim that $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$. To see this, note that $f(\langle M \rangle) = \langle M' \rangle \in \text{REGULAR}_{\text{TM}}$ iff $A(M')$ is regular. We claim that $A(M')$ is regular iff $\langle M \rangle \in A(M)$. To see this, note that if $\langle M \rangle \in A(M)$, then $M'$ accepts all strings, either because that string is of the form $0^n1^n$ or because $M$ eventually accepts $\langle M \rangle$. Thus $A(M') = \Sigma^*$, which is regular. Otherwise, if $\langle M \rangle \notin A(M)$, then $M'$ only accepts strings of the form $0^n1^n$, so $A(M) = \{ 0^n1^n | n \in \mathbb{N} \}$, which is not regular. Finally, $\langle M \rangle \in A(\langle M \rangle)$ iff $\langle M \rangle \in \overline{L}_D$. Thus $\langle M \rangle \in \overline{L}_D$ iff $f(\langle M \rangle) \in \text{REGULAR}_{\text{TM}}$, so $f$ is a mapping reduction from $\overline{L}_D$ to $\text{REGULAR}_{\text{TM}}$. Therefore, $\overline{L}_D \leq_M \text{REGULAR}_{\text{TM}}$. ■
The Limits of Computability

\[
\begin{align*}
\text{REGULAR}_{\text{TM}} & \quad \text{REGULAR}_{\text{TM}} \\
\text{HALT} & \quad \text{HALT} \\
L_D & \quad L_D \\
A_{\text{TM}} & \quad A_{\text{TM}} \\
\text{ONES}_{\text{TM}} & \quad \text{ONES}_{\text{TM}} \\
\text{ONLYONES}_{\text{TM}} & \quad \text{ONLYONES}_{\text{TM}} \\
0^*1^* & \quad 0^*1^* \\
\text{DOGWALK} & \quad \text{DOGWALK} \\
\text{ADD} & \quad \text{ADD} \\
\text{co-RE} & \quad \text{RE} \\
\text{RE} & \quad \text{RE} \\
\text{All Languages} & \quad \text{All Languages}
\end{align*}
\]
Beyond $\text{RE}$ and $\text{co-RE}$

• The most famous problem that is neither $\text{RE}$ nor $\text{co-RE}$ is the TM equality problem:

$$\text{EQ}_{\text{TM}} = \{ \langle M_1, M_2 \rangle \mid \mathcal{L}(M_1) = \mathcal{L}(M_2) \}$$

• This is why we have to write testing code; there's no way to have a computer prove or disprove that two programs always have the same output.

• See Sipser for a proof of this fact.
Why All This Matters
The Limits of Computability

- RE
- co-RE
  - $\text{REGULAR}_{TM}$
  - $\text{HALT}_{TM}$
  - $\text{ONES}_{TM}$
  - $\text{ONLYONES}_{TM}$
- RE
  - $\text{REGULAR}_{TM}$
  - $\text{HALT}_{TM}$
  - $\text{ONES}_{TM}$
  - $\text{ONLYONES}_{TM}$
- co-RE
  - $\text{REGULAR}_{TM}$
  - $\text{HALT}_{TM}$
  - $\text{ONES}_{TM}$
  - $\text{ONLYONES}_{TM}$

Languages:
- $0^*1^*$
- DOGWALK
- ADD

All Languages
What problems can be solved a computer?
What problems can be solved **efficiently** a computer?
Where We've Been

- The class \( R \) represents problems that can be solved by a computer.
- The class \( \text{RE} \) represents problems where answers can be verified by a computer.
- The class \( \text{co-RE} \) represents problems where answers can be refuted by a computer.
- The mapping reduction can be used to find connections between problems.
Where We're Going

- The class \( \mathbf{P} \) represents problems that can be solved \textit{efficiently} by a computer.
- The class \( \mathbf{NP} \) represents problems where answers can be verified \textit{efficiently} by a computer.
- The class co-\( \mathbf{NP} \) represents problems where answers can be \textit{efficiently} refuted by a computer.
- The \textit{polynomial-time} mapping reduction can be used to find connections between problems.
Next Time

- **Introduction to Complexity Theory**
  - What problems can be solved *efficiently*?
  - How do you define efficiency?
  - How do you measure it?
  - What tools will we need?