The Limits of Computation

28 April 2020
Where are we?
Self-referential programs

It is possible to build Turing machines that get their own encodings and perform arbitrary computations on them.
What does this program do?

uh-oh.py:

def will_accept(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("uh-oh.py").read()
    if will_accept(my_source, my_input):
        return False
    else:
        return True
What does this program do?

**uh-oh.py:**

```python
def will_accept(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("uh–oh.py").read()
    if will_accept(my_source):
        return False
    else:
        return True
```

*The program has its own source code stored in the variable my_source.*
What does this program do?

`uh-oh.py:`

```python
def will_accept(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("uh-oh.py").read()
    if will_accept(my_source, my_input):
        return False
    else:
        return True
```

*It now asks – am I going to accept my input?*
What does this program do?

`uh-oh.py`:

```python
def will_accept(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("uh-oh.py").read()
    if will_accept(my_source, my_input):
        return False
    else:
        return True
```

If this program accepts its input, then it rejects the input!

If this program doesn’t accept its input, then it accepts the input!
What does this program do?

**uh-oh.py:**

```python
def will_accept(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("uh-oh.py").read()
    if will_accept(my_source, my_input):
        return False
    else:
        return True
```

A self-defeating object

Using that object against itself
What does this program do?

**uh-oh.py:**

```python
def will_accept(program, input):
    ...some implementation...

    "The largest integer, n"

def main(my_input):
    my_source = open("uh-oh.py").read()

    if will_accept(my_source, my_input):
        return False
    else:
        return True

    "The number n + 1"
```
Regular languages

Context-free languages

All languages

$R$

$R E$

$A_{TM}$

$HALT_{TM}$
Ramifications, or: So what?
These problems might seem really convoluted and not very exciting, so who cares if we can’t solve them?

The same line of reasoning we used to show that it’s undecidable

whether a Turing machine will accept its input ($A_{TM}$) or
whether it will halt on its input ($HALT$)

can be used to show many important, practical problems are impossible to solve.
Secure voting

Suppose you want to make a voting machine for use in an imaginary election between two parties.¹

Let \( \Sigma = \{r, d\} \), for no particular reason.

A string consists of a series of votes for the candidates.

For example, \( rrddd\text{dr}d \) means “two people voted for \( r \), then three people voted for \( d \), then one more person voted for \( r \), then one more person voted for \( d \)”.  

¹ We live in a two-party system. Sorry.
A voting machine is a program that takes as input a string of rs and ds and then reports whether person r won the election.

It would be equivalent to ask “did d win”?

For simplicity, this model assumes centralized voting, e.g., done online.

**Question**: Given a Turing machine that someone claims is a secure voting machine, could we automatically check whether it’s really a secure voting machine?
Watch what happens when I vote for Bobby Newport.

But watch what happens when you vote for me.

Official Sweetums Voting Machine

Good Choice!
Enjoy a voucher for a complimentary Sweetums candy bar.

Are you sure?
Take a second and think it over...

[ buzzer sounds ]
def main(w):
    r_votes = count_rs(w)
    d_votes = count_ds(w)
    if r_votes > d_votes:
        return True  # Rs won
    else:
        return False  # Rs lost

A (simple) secure voting machine

def main(w):
    if w[0] == "r":
        return True  # Rs won
    else:
        return False  # Rs lost

A (simple) insecure voting machine
An (evil) insecure voting machine

```python
def main(w):
    r_votes = count_rs(w)
    d_votes = count_ds(w)
    if r_votes == d_votes:
        # Tied; Rs lose.
        return False
    if r_votes < d_votes:
        # Ds win; Rs lose.
        return False
    else:
        # Rs win.
        return True
```
def main(w):
    r_votes = count_rs(w)
    d_votes = count_ds(w)
    if r_votes == d_votes:
        # Tied; Rs lose.
        return False
    if r_votes < d_votes:
        # Ds win; Rs lose.
        return False
    else:
        # Rs win.
        return True

This is assignment; == is equality testing. Python won’t actually let you do this, but other languages like C or Java will!
```python
def main(w):
    n = len(w)
    while n > 1:
        if n % 2 == 0:
            n /= 2
        else:
            n = 3 * n + 1
    r_votes = count_rs(w)
    d_votes = count_ds(w)
    if r_votes > d_votes:
        return True  # Rs won
    else:
        return False  # Rs lose
```

def main(w):
    n = len(w)
    while n > 1:
        if n % 2 == 0:
            n /= 2
        else:
            n = 3 * n + 1
    r_votes = count_rs(w)
    d_votes = count_ds(w)
    if r_votes > d_votes:
        return True  # Rs won
    else:
        return False  # Rs lose

No one knows!
Secure voting

The secure voting problem is the following:

Given a TM \( M \), is the language of \( M \)

\[ \{ w \in \Sigma^* \mid w \text{ has more } \text{rs} \text{ than } \text{ds} \} \]?

**Claim**: This problem is not decidable; there is no algorithm that can check an arbitrary TM to verify that it’s a secure voting machine!

A program that decides whether arbitrary input programs are secure voting machines is self-defeating. It therefore doesn’t exist.
A decider for secure voting

Suppose that, somehow, we managed to build a decider for the secure voting problem:

We could represent this in software as a procedure

\[\text{is\_secure\_vm(} \text{program} \text{)}\]
def is_secure_vm(program):
    ...some implementation...

def main(w):
    my_source = open("vm.py").read()
    answer = count_rs(w) > count_ds(w)
    if is_secure_vm(my_source):
        return not answer
    else:
        return answer
What happens if this program is a secure voting machine?

vm.py:

```python
def is_secure_vm(program):
    ...some implementation...

def main(w):
    my_source = open("vm.py").read()
    answer = count_rs(w) > count_ds(w)
    if is_secure_vm(my_source):
        return not answer
    else:
        return answer
```

Then it’s not a secure voting machine!
What happens if this program is not a secure voting machine?

vm.py:

def is_secure_vm(program):
    ...some implementation...

def main(w):
    my_source = open("vm.py").read()
    answer = count_rs(w) > count_ds(w)
    if is_secure_vm(my_source):
        return not answer
    else:
        return answer

Then it’s a secure voting machine!
Interpreting this result

This tells us that there is no general algorithm that we can follow to determine whether a program is a secure voting machine.

In other words, any general algorithm to check voting machines will always be wrong on at least one input.

The previous example might seem contrived, but it’s not. This is a problem we really would like to be able to solve — but it’s provably impossible!
What can we do?

Design algorithms that work in *many* but not *all* cases.

This is often done in practice!

Fall back on human verification of voting machines.

We do this too!

Carry a healthy degree of skepticism about electronic voting machines.

We were born skeptical.
Wrapping up undecidability
We’ve seen a general pattern in proving undecidability (i.e., non-membership in $\mathbb{R}$):

Assume the language in question – usually a language about TMs – is decidable.

Build a machine that decides whether it has the property, then chooses to do something contrary to that property.

Conclude that something is terribly wrong, meaning that the original language wasn’t decidable.
Two intuitions

The *avoid your fate* intuition:

Construct a machine so that it learns its fate (i.e., *decides* what to do next), then actively chooses to do the opposite.

The *impossible bind* intuition:

Imagine the TM in conversation with the decider that can allegedly predict what happens next.

Have the TM tell the decider that it’s going to do the opposite of whatever the decider says.

The decider’s in an impossible bind – anything it says must be wrong!
Beyond R and RE
Beyond R and RE

We’ve seen how to use self-reference to show undecidability – there are languages not in R.

It’s time to break out of RE.
Background: Cantor’s diagonalization method
Consider the sets:

\[ \mathbb{N}_0 = \{0, 1, 2, 3, \ldots\} \]

i.e., the natural numbers including 0

\[ S = \{n \mid n \in \mathbb{N}_0 \text{ and } n \text{ is even}\} \]

Is \( \mathbb{N}_0 \) bigger than \( S \), i.e., is \(|\mathbb{N}_0| > |S|\)?
By definition, two sets have the same size if there is a way to pair their elements off without leaving any elements uncovered.
By definition, two sets have the same size if there is a way to pair their elements off without leaving any elements uncovered.

Everything’s been paired up, but this one’s all alone 😞
Infinite cardinalities

\[ \mathbb{N}: \{0, 1, 2, 3, 4, 5, 6, 7, 8, \ldots\} \]

\[ S: \{0, 2, 4, 6, 8, \ldots\} \]

\[ S = \{n \mid n \in \mathbb{N}_0 \text{ and } n \text{ is even}\} \]

Two sets have the same size if there is a way to pair their elements off without leaving any elements uncovered.
Infinite cardinalities

\[ \mathbb{N} : \{0, 1, 2, 3, 4, 5, 6, 7, 8, \ldots \} \]

\[ S : \{0, 2, 4, 6, 8, \ldots \} \]

\[ S = \{n \mid n \in \mathbb{N}_0 \text{ and } n \text{ is even} \} \]

Two sets have the same size if *there is some way* to pair their elements off without leaving any elements uncovered.
Infinite cardinalities

\[ \mathbb{N}: \{0, 1, 2, 3, 4, 5, 6, 7, 8, \ldots\} \]

\[ S: \{0, 2, 4, 6, 8, 10, 12, 14, 16, \ldots\} \]

\[ S = \{n \mid n \in \mathbb{N}_0 \text{ and } n \text{ is even}\} \]

\[ n \leftrightarrow 2n \]

Counterintuitively, there are as many even natural numbers as there are natural numbers!
What about the set of all integers ($\mathbb{Z}$) and natural numbers ($\mathbb{N}_0$)?

$\mathbb{Z}$ is infinite in two directions!
What about the set of all integers (\(\mathbb{Z}\)) and natural numbers (\(\mathbb{N}_0\))? 

\(\mathbb{Z}\) is infinite in two directions!

\textbf{It can be done! Pair the positive integers in \(\mathbb{Z}\) with even natural numbers and the negative integers in \(\mathbb{Z}\) with odd natural numbers!}
Do all infinite sets have the same cardinality?
Consider: If $|S|$ is infinite, what is the relation between $|S|$ and $|\mathcal{P}(S)|$, where $\mathcal{P}$ denotes the power set?

Does $|S| = |\mathcal{P}(S)|$?
If $|S| = |\mathcal{P}(S)|$, then we can pair up the elements of $\mathcal{P}(S)$ and the elements of $S$ itself without leaving anything out.

What would that look like?
\[ x_0 \leftrightarrow \{ x_0, x_2, x_4, \ldots \} \]
\[ x_1 \leftrightarrow \{ x_3, x_5, \ldots \} \]
\[ x_2 \leftrightarrow \{ x_0, x_1, x_2, x_5, \ldots \} \]
\[ x_3 \leftrightarrow \{ x_1, x_4, \ldots \} \]
\[ x_4 \leftrightarrow \{ x_2, \ldots \} \]
\[ x_5 \leftrightarrow \{ x_0, x_4, x_5, \ldots \} \]
\[ \ldots \leftrightarrow \{ \ldots \} \]
\[
\begin{array}{cccccc}
  x_0 & x_1 & x_2 & x_3 & x_4 & x_5 & \ldots \\
\end{array}
\]

\[
x_0 \leftrightarrow \{ x_0, x_2, x_4, \ldots \}
\]

\[
x_1 \leftrightarrow \{ x_3, x_5, \ldots \}
\]

\[
x_2 \leftrightarrow \{ x_0, x_1, x_2, x_5, \ldots \}
\]

\[
x_3 \leftrightarrow \{ x_1, x_4, \ldots \}
\]

\[
x_4 \leftrightarrow \{ x_2, \ldots \}
\]

\[
x_5 \leftrightarrow \{ x_0, x_4, x_5, \ldots \}
\]

\[
\ldots \leftrightarrow \{ \ldots \}
\]
\[
\begin{array}{cccccc}
\begin{array}{c}
\times_0 \\
\times_1 \\
\times_2 \\
\times_3 \\
\times_4 \\
\times_5 \\
\ldots
\end{array}
& \begin{array}{c}
\times_0 \\
\times_2 \\
\times_0 \\
\times_1 \\
\times_2 \\
\times_0 \\
\ldots
\end{array}
& \begin{array}{c}
\times_2 \\
\times_3 \\
\times_2 \\
\times_4 \\
\times_2 \\
\times_0 \\
\ldots
\end{array}
& \begin{array}{c}
\times_4 \\
\times_5 \\
\times_5 \\
\times_4 \\
\times_5 \\
\times_4 \\
\ldots
\end{array}
& \begin{array}{c}
\ldots \\
\ldots \\
\ldots \\
\ldots \\
\ldots \\
\ldots \\
\ldots
\end{array}

\end{array}
\]
Which element is paired with this set?
Which element is paired with this set?
<table>
<thead>
<tr>
<th></th>
<th>$x_0$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$x_5$</th>
<th>$\ldots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_0$</td>
<td>{ $x_0$, }</td>
<td>$x_2$,</td>
<td>$x_4$,</td>
<td></td>
<td></td>
<td></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_1$</td>
<td>{ }</td>
<td>$x_3$,</td>
<td>$x_5$,</td>
<td></td>
<td></td>
<td></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>{ $x_0$, }</td>
<td>$x_2$,</td>
<td>$x_5$,</td>
<td></td>
<td></td>
<td></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>{ }</td>
<td>$x_1$,</td>
<td>$x_4$,</td>
<td></td>
<td></td>
<td></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_4$</td>
<td>{ }</td>
<td>$x_2$,</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$x_5$</td>
<td>{ $x_0$, }</td>
<td>$x_4$,</td>
<td>$x_5$,</td>
<td></td>
<td></td>
<td></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>{ }</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>{ }</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

“Flip” which elements are included
<table>
<thead>
<tr>
<th></th>
<th>$x_0$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$x_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_0$ $\leftrightarrow$</td>
<td>${x_0, x_2, x_4, \ldots}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_1$ $\leftrightarrow$</td>
<td>${x_3, x_5, \ldots}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_2$ $\leftrightarrow$</td>
<td>${x_0, x_2, x_5, \ldots}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_3$ $\leftrightarrow$</td>
<td>${x_1, x_4, \ldots}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_4$ $\leftrightarrow$</td>
<td>${x_2, \ldots}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_5$ $\leftrightarrow$</td>
<td>${x_0, x_4, x_5, \ldots}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\ldots$ $\leftrightarrow$</td>
<td>${\ldots }$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Which element is paired with this set?
Which element is paired with this set?

It differs from every set in at least one position!

Which element is paired with this set?
The diagonalization proof

No matter how we pair up elements of S and subsets of S, the complemented diagonal won’t appear in the table.

In row $n$, the $n$th element must be wrong.

No matter how we pair up elements of S and subsets of S, there is always at least one subset left over.

The result is Cantor’s Theorem: Every set is strictly smaller than its power set.

If $S$ is a set, then $|S| < |\mathcal{P}(S)|$. 
What does this have to do with computation?

Consider:

“The set of all Turing machines (computer programs)”

“The set of all languages (problems to solve)”
We can use diagonalization to show that there must be languages that are non-recognizable because there are more distinct languages than there are Turing machines to recognize them.

The full proof of this is in the book.

We’ll move on to using diagonalization to show that there is a specific language, $L_D$, that is not recognizable.
Languages, TMs, and TM encodings

**Recall**: The language of a TM $M$ is the set

$$L(M) = \{ w \in \Sigma^* \mid M \text{ accepts } w \}$$

Some of the strings in this set might be descriptions of TMs.

What happens if we list off all Turing machines, looking at how those TMs behave given other TMs as input?
\[ \mathcal{M}_0 \]
\[ \mathcal{M}_1 \]
\[ \mathcal{M}_2 \]
\[ \mathcal{M}_3 \]
\[ \mathcal{M}_4 \]
\[ \mathcal{M}_5 \]
\[ \ldots \]
All Turing machines, listed in some order
All descriptions of Turing machines, listed in the same order

<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>$\cdots$</th>
</tr>
</thead>
<tbody>
<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>(\cdots)</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>(\cdots)</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>(\cdots)</td>
</tr>
<tr>
<td>(M_1)</td>
<td>(M_2)</td>
<td>(M_3)</td>
<td>(M_4)</td>
<td>(M_5)</td>
<td>(\cdots)</td>
<td></td>
</tr>
<tr>
<td>(M_2)</td>
<td>(M_3)</td>
<td>(M_4)</td>
<td>(M_5)</td>
<td>(\cdots)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(M_3)</td>
<td>(M_4)</td>
<td>(M_5)</td>
<td>(\cdots)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(M_4)</td>
<td>(M_5)</td>
<td>(\cdots)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(M_5)</td>
<td>(\cdots)</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>$M_2$</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></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></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>$\cdots$</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></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>$\langle M_0 \rangle$</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>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</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>$\cdots$</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>$\cdots$</td>
</tr>
<tr>
<td>$M_1$</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$M_2$</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$M_3$</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$M_4$</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$M_5$</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</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>
</tbody>
</table>
| $M_0$           | Acc             | No              | No              | Acc             | Acc             | No  | ...
| $M_1$           | Acc             | Acc             | Acc             | Acc             | Acc             | Acc  | ...
| $M_2$           | Acc             | Acc             | Acc             | Acc             | Acc             | Acc  | ...
| $M_3$           | No              | Acc             | Acc             | No              | Acc             | Acc  | ...
| $M_4$           | Acc             | No              | Acc             | No              | Acc             | No   | ...
| $M_5$           | No              | No              | Acc             | Acc             | No              | No   | ...
| ...             | ...             | ...             | ...             | ...             | ...             | ...  | ...
<table>
<thead>
<tr>
<th></th>
<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>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</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>
<td></td>
</tr>
<tr>
<td>---</td>
<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>
<td>...</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>
<td>...</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>
<td>...</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>
<td>...</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>
<td>...</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>
<td>...</td>
</tr>
<tr>
<td>...</td>
<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>
<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>
<td>...</td>
</tr>
<tr>
<td>------</td>
<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>
<td>...</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>
<td>...</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>
<td>...</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>
<td>...</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>
<td>...</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>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Flip all “accept”s to “no”s and vice versa.
What TM has this behavior?
<table>
<thead>
<tr>
<th></th>
<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>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\ldots$</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>
<td>$\cdots$</td>
</tr>
<tr>
<td>---</td>
<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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</td>
</tr>
<tr>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$\langle M_0 \rangle$</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\cdots$</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>$\cdots$</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</td>
</tr>
<tr>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>$\cdots$</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\cdots$</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>
<td>$\ldots$</td>
</tr>
<tr>
<td>-----</td>
<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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</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>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\ldots$</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>
<td></td>
</tr>
<tr>
<td>-------</td>
<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>
<td></td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>...</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>
<td></td>
</tr>
<tr>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>...</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>
<td></td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</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>$\cdots$</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</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>
<td>$\cdots$</td>
</tr>
<tr>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\cdots$</td>
<td>$\cdots$</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>(\cdots)</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</td>
</tr>
<tr>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>(\cdots)</td>
</tr>
</tbody>
</table>

No TM has this behavior!
<table>
<thead>
<tr>
<th></th>
<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>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</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>
<td>(\cdots)</td>
</tr>
<tr>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
<td>(\cdots)</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>(\cdots)</td>
</tr>
</tbody>
</table>

“The language of all TMs that do not accept their own descriptions”
\{ \langle M \rangle \mid M \text{ is a TM that does not accept } \langle M \rangle \}\}
The *diagonalization language*, denoted $L_D$, is defined as

$$L_D = \{\langle M \rangle \mid M \text{ is a TM and } M \text{ does not accept } \langle M \rangle \}$$

We constructed this language to be different from the language of every TM.

Therefore, $L_D \not\in \text{RE}$! Let’s prove this.
THEOREM. $L_D \notin \text{RE}$.

PROOF. By contradiction; assume that $L_D \in \text{RE}$. This means that there is a TM $R$ such that $L(R) = L_D$.

Since $R$ is a recognizer for $L_D$, we see that $R$ accepts $\langle R \rangle$ if and only if $\langle R \rangle \in L_D$.

By the definition of $L_D$, we know that $\langle R \rangle \in L_D$ if and only if $R$ does not accept $\langle R \rangle$.

Combining the two statements tell us that $R$ accepts $\langle R \rangle$ if and only if $R$ doesn’t accept $\langle R \rangle$.

We’ve reached a contradiction, so our assumption was wrong and $L_D \notin \text{RE}$. ■
Classes of languages
All languages
All languages
There is a TM $M$ that rejects $w$ iff $w \not\in L$.

There is a TM $M$ that accepts $w$ iff $w \in L$.

All languages
There is a TM $M$ that accepts $w$ iff $w \in L$

There is a TM $M$ that rejects $w$ iff $w \notin L$
There is a TM $M$ that rejects $w$ iff $w \notin L$.

There is a TM $M$ that accepts $w$ iff $w \in L$.
A new class of languages

A language $L$ is in $\text{RE}$ iff there is a TM $M$ such that

- if $w \in L$, then $M$ accepts $w$
- if $w \notin L$, then $M$ does not accept $w$

A TM $M$ of this sort is called a **recognizer**, and $L$ is called $\text{Turing-recognizable}$.

A language $L$ is in $\text{co-RE}$ iff there is a TM such that

- if $w \in L$, then $M$ does not reject $w$
- if $w \notin L$, then $M$ rejects $w$

A TM $M$ of this sort is called a **co-recognizer** and the language $L$ is called $\text{co-Turing-recognizable}$.
RE and co-RE

Intuitively, \textbf{RE} consists of all problems where a TM can exhaustively search for \textit{proof} that \( w \in L \).

- If \( w \in L \), the TM will find the proof.
- If \( w \notin L \), the TM cannot find a proof.

Intuitively, \textbf{co-RE} consists of all problems where a TM can exhaustively search for \textit{disproof} that \( w \in L \).

- If \( w \in L \), the TM cannot find the disproof.
- If \( w \notin L \), the TM will find the disproof.
RE and co-RE languages

$A_{TM}$ is an RE language:

- Simulate the TM $M$ on the string $w$.
- If you find that $M$ accepts $w$, accept.
- If you find that $M$ rejects $w$, reject.
- (If $M$ loops, we implicitly loop forever.)

$\overline{A_{TM}}$ is a co-RE language:

- Simulate the TM $M$ on the string $w$.
- If you find that $M$ accepts $w$, reject.
- If you find that $M$ rejects $w$, accept.
- (If $M$ loops, we implicitly loop forever.)
## RE and co-RE languages

$L_D$ is a co-RE language:

Simulate the TM $M$ on $\langle M \rangle$.
If you find that $M$ accepts $\langle M \rangle$, reject.
If you find that $M$ rejects $\langle M \rangle$, accept.
(If $M$ loops, we implicitly loop forever.)

$L_D$ is an RE language:

Simulate the TM $M$ on $\langle M \rangle$.
If you find that $M$ accepts $\langle M \rangle$, accept.
If you find that $M$ rejects $\langle M \rangle$, reject.
(If $M$ loops, we implicitly loop forever.)
There is a TM \( M \) that accepts \( w \) iff \( w \in L \)

There is a TM \( M \) that rejects \( w \) iff \( w \notin L \)
There is a TM $M$ that accepts $w$ iff $w \in L$

There is a TM $M$ that rejects $w$ iff $w \notin L$
There is a TM $M$ that accepts $w$ iff $w \in L$

There is a TM $M$ that rejects $w$ iff $w \notin L$
There is a TM $M$ that accepts $w$ iff $w \in L$.

There is a TM $M$ that rejects $w$ iff $w \notin L$.

All languages
**RE and co-RE**

**THEOREM.** $L \in \text{RE}$ iff $\overline{L} \in \text{co-RE}$.

**PROOF SKETCH.** Start with a recognizer $M$ for $L$. Then, flip its accepting and rejecting states to make machine $M'$. Then

<table>
<thead>
<tr>
<th>$M'$ rejects $w$</th>
<th>$M'$ does not reject $w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\iff M$ accepts $w$</td>
<td>$\iff M'$ accepts $w$ or $M'$ loops on $w$</td>
</tr>
<tr>
<td>$\iff w \in L$</td>
<td>$\iff M$ rejects $w$ or $M$ loops on $w$</td>
</tr>
<tr>
<td>$\iff w \not\in \overline{L}$</td>
<td>$\iff w \not\in L$</td>
</tr>
</tbody>
</table>

The same approach works if we flip the accept and reject states of a co-recognizer for $\overline{L}$. ■
**R, RE, and co-RE**

Every language in \( R \) is in both \( RE \) and \( co-RE \).

Why?

A decider for \( L \) accepts all \( w \in L \) and rejects all \( w \notin L \).

In other words, \( R \subseteq (RE \cap co-RE) \).

**Question:** Does \( R = (RE \cap co-RE) \)?
There is a TM $M$ that rejects $w$ iff $w \notin L$.

There is a TM $M$ that accepts $w$ iff $w \in L$.

All languages
Which picture is correct?

There is a TM $M$ that accepts $w$ iff $w \in L$

There is a TM $M$ that rejects $w$ iff $w \notin L$

<table>
<thead>
<tr>
<th>co-RE</th>
<th>RE</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{TM}$</td>
<td>$A_{TM}$</td>
</tr>
<tr>
<td>HALT$_{TM}$</td>
<td>HALT$_{TM}$</td>
</tr>
<tr>
<td>$L_D$</td>
<td>$L_D$</td>
</tr>
<tr>
<td>${a^nb^nc^n}$</td>
<td></td>
</tr>
</tbody>
</table>

All languages
R, RE, and co-RE

THEOREM. If $L \in \text{RE}$ and $L \in \text{co-RE}$, then $L \in \text{R}$.

PROOF SKETCH. Since $L \in \text{RE}$, there is a recognizer $M$ for it. Since $\overline{L} \in \text{co-RE}$, there is a co-recognizer $\overline{M}$ for it. Then this TM $D$ is a decider for $L$:

\[
D = \text{“On input } w : \\
1. \text{ Run } M \text{ on } w \text{ and } \overline{M} \text{ on } w \text{ in parallel by alternating simulating one step of } M \text{ and one step of } \overline{M}. \\
2. \text{ If } M \text{ accepts } w, \text{ accept.} \\
3. \text{ If } \overline{M} \text{ rejects } w, \text{ reject.”}
\]
There is a TM $M$ that accepts $w$ iff $w \in L$

There is a TM $M$ that rejects $w$ iff $w \notin L$

This picture is correct!
We know $A_{TM}$ is Turing-recognizable.

If $A_{TM}$ were Turing-recognizable, then $A_{TM}$ would be decidable.

But we have proved $A_{TM}$ is not decidable.

So $A_{TM}$ must not be Turing-recognizable (i.e., it is non-Turing-recognizable).

*If you know a language is not decidable, you can easily prove its complement is non-Turing-recognizable.*
Acknowledgments

This lecture incorporates material from:

W. Daniel Hillis
Nancy Ide
Keith Schwarz
Michael Sipser