The Limits of Computation

Lecture 22

3 December 2019
Exam 2

Not graded yet! Too much pie.

Assignment 7

Corrections due

Assignment 8

Last one – out today

Extra credit programming assignment

Out – due last possible day
Where are we?
We’ve introduced our final model of a computer, the Turing machine.

We designed Turing machines at the level of individual states and transitions.

In your reading (and on Assignment 7), you’ve seen that you can also describe a Turing machine more abstractly, like writing pseudo-code.
The *Church–Turing thesis* states:

Every effective method of computation is either equivalent to or weaker than a Turing machine.
A language $L$ is **Turing-recognizable** if there is a TM $M$ such that

$$\forall w \in \Sigma^*. M \text{ accepts } w \iff w \in L.$$ 

This is a “weak” notion of solving a problem:

- If $w \in L$, then $M$ accepts $w$.
- If $w \notin L$, then $M$ does not accept $w$.
  
  It might reject or it might loop forever.

The class $\textbf{RE}$ consists of all Turing-recognizable languages.
A language $L$ is **Turing-decidable** if there is a TM $M$ such that

$$\forall w \in \Sigma^*. (M \text{ accepts } w \iff w \in L) \land (M \text{ halts on all inputs}).$$

This is a “strong” notion of solving a problem:

- If $w \in L$, then $M$ accepts $w$.
- If $w \notin L$, then $M$ rejects $w$.

The class $R$ consists of all Turing-decidable languages.
If \( \text{Obj} \) is an object, then \( \langle \text{Obj} \rangle \) denotes some string representing \( \text{Obj} \), similar to how it might be stored on disk or in memory on a real computer.

We can encode multiple objects as a single string. For example, if \( M \) is a TM and \( w \) is a string, then \( \langle M, w \rangle \) is a string representing the pair of \( M \) and \( w \).
There is a TM named \( U \) that is a *universal Turing machine*.

\( U \) takes as input a pair \( \langle M, w \rangle \), where \( M \) is a TM and \( w \) is a string.

\( U \) does to \( \langle M, w \rangle \) whatever \( M \) does to \( w \).
The language of $U$ is called $A_{TM}$:

$$A_{TM} = L(U) = \{\langle M, w \rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

$A_{TM}$ is the acceptance language for Turing machines. Because there is a Turing machine, $U$, that recognizes $A_{TM}$, we know $A_{TM} \in RE$. 
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:**

```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
```
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!
Today we’ll look at some other undecidable problems and think about what this means for our ability to compute.
The Halting Problem
The most famous undecidable problem is the *Halting Problem*, which asks:

Given a Turing machine $M$ and a string $w$, will $M$ halt when run on $w$?

or, as a language,

$$HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM that halts on } w \}$$

This is an **RE** language. *(We’ll see why later.)*

How do we know that it’s undecidable?
Claim: A decider for $\textit{HALT}_{TM}$ is a self-defeating object. Therefore it doesn’t exist.
A decider for $\text{HALT}^{\text{TM}}$

Suppose we managed to build a decider for $\text{HALT}^{\text{TM}} = \{\langle M, w \rangle \mid M \text{ is a TM that halts on } w\}$:

We could represent this in software as a procedure `will_halt(program, input)`.
def will_halt(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("halt.py").read()
    if will_halt(my_source, my_input):
        while True:  # Infinite loop
            pass
    else:
        return True
Try running this on any input.

halt.py:

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

def main(my_input):
  my_source = open("halt.py").read()
  if will_halt(my_source, my_input):
    while True: # Infinite loop
      pass
  else:
    return True
What happens if the program *halts on its input*?

**halt.py:**

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

def main(my_input):
    my_source = open("halt.py").read()
    if will_halt(my_source, my_input):
        while True: # Infinite loop
            pass
    else:
        return True
```
def will_halt(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("halt.py").read()
    if will_halt(my_source, my_input):
        while True:
            # Infinite loop
            pass
    else:
        return True
What happens if the program loops on its input?

**halt.py:**

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

def main(my_input):
    my_source = open("halt.py").read()
    if will_halt(my_source, my_input):
        while True:  # Infinite loop
            pass
    else:
        return True
```
What happens if the program loops on its input?

**halt.py:**

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

def main(my_input):
    my_source = open("halt.py").read()
    if will_halt(my_source, my_input):
        while True:  # Infinite loop
            pass
    else:
        return True  # It halts on the input!
```

**halt.py:** What happens if the program loops on its input?

It halts on the input!
def will_halt(program, input):
    ...some implementation...

def main(my_input):
    my_source = open("halt.py").read()
    if will_halt(my_source, my_input):
        while True:  # Infinite loop
            pass
    else:
        return True

halt.py:
The self-defeating object
Using that object against itself
**Theorem:** $HALT_{TM} \not\in R$.

**Proof:** By contradiction; assume that $HALT_{TM} \in R$. Then there is some decider $D$ for $HALT_{TM}$, which we can represent in software as a procedure `will_halt` that takes as input the source code of a program and an input, and returns true if the program halts on the input and false otherwise.

Given this, we could construct this program $P$:

```python
if will_halt(my_source, my_input):
    while True: pass
else:
    return True
```

Choose any string $w$ and trace through the execution of program $P$ on input $w$: If `will_halt(my_source, my_input)` returns true, this means that $P$ must halt on its input $w$, but instead it loops on it. Otherwise, if `will_halt(my_source, my_input)` returns false, this means that $P$ must not halt on its input $w$, but instead it halts and accepts it.

In both cases, we reach a contradiction, so our assumption must have been wrong. Therefore, $HALT_{TM} \not\in R$. ■
**Theorem:** \( \text{HALT}_{\text{TM}} \notin R \).

**Proof:** By contradiction; assume that \( \text{HALT}_{\text{TM}} \in R \). Then there is some decider \( D \) for \( \text{HALT}_{\text{TM}} \). If this machine is given any TM/string pair, it will then determine whether the TM halts on the string and report back the answer.

Given this, we could construct the following TM:

\[
M = \text{"On input } w:\n1. \text{Have } M \text{ obtain its own description } \langle M \rangle.
2. \text{Run } D \text{ on } \langle M, w \rangle \text{ and see what it says.}
3. \text{If } D \text{ says that } M \text{ halts on } w, \text{ go into an infinite loop.}
4. \text{If } D \text{ says that } M \text{ loops on } w, \text{ accept."
}\]

Choose any string \( w \) and trace through the execution of the machine, focusing on the answer given back by machine \( D \). If \( D \) says that \( M \) will halt on \( w \), notice that \( M \) then proceeds to loop on \( w \), contradicting what \( D \) says. Otherwise, if \( D \) says that \( M \) will loop on \( w \), notice that \( M \) then proceeds to accept \( w \), so \( M \) halts on \( w \), contradicting what \( D \) says.

In both cases, we reach a contradiction, so our assumption must have been wrong. Therefore, \( \text{HALT}_{\text{TM}} \notin R \). □

*Same proof, without using the equivalence of TMs and code*
CLAIM: $HALT_{TM} \in \text{RE}$.

IDEA: If you were certain that a Turing machine $M$ halted on a string $w$, could you convince me of that?

Yes – just run $M$ on $w$ and see what happens!
Moral: This isn’t necessarily Microsoft’s fault.
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.

E.g., $rrddddr$ 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. It sucks, but accept it and vote for the lesser of two evils.
A voting machine is a program that takes as input a string of $r$s and $d$s and then reports whether person $r$ won the election.

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?
To cancel your vote and select someone better, press the Sweetums logo.

- [computer voice]
  TO CANCEL YOUR VOTE AND SELECT

- I'M SORRY.
  I JUST DON'T SEE THE PROBLEM.
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 # Ds won

A (simple) secure voting machine

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

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:
        # So close; we can pretend Ds won
        return False
    if r_votes < d_votes:
        # Ds won fair and square
        return False
    else:
        # Rs won fair and square
        return True
```
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 # Ds won
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 # Ds won

No one knows!
Secure voting

The *secure voting problem* is the following:

Given a TM $M$, is the language of $M \{w \in \Sigma^* \mid w$ has more rs than 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

\[ \text{is\_secure\_vm}(\text{program}) \]
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
    return answer

What happens if this program is a secure voting machine?
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
    return answer
```

Then it's not a secure voting machine!
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
    return answer

vm.py:

What happens if this program is 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
    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
Regular languages

Context-free languages

$\text{RE}$

$\text{R}$

Anything out here?
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 \( |\mathbb{N}_0| > |S| \)?
{lion, dog, koala, pig}
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 \}

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

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

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

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

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

\[ n \leftrightarrow 2n \]

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

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

$\mathbb{Z}$ is infinite in two directions!

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

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

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

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

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

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

\[ \ldots \leftrightarrow \{ \ldots \} \]
<table>
<thead>
<tr>
<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>...</th>
</tr>
</thead>
</table>

$x_0 \leftrightarrow \{ x_0, x_2, x_4, \ldots \}$

$x_1 \leftrightarrow \{ x_3, x_5, \ldots \}$

$x_2 \leftrightarrow \{ x_0, x_1, x_2, x_5, \ldots \}$

$x_3 \leftrightarrow \{ x_1, x_4, \ldots \}$

$x_4 \leftrightarrow \{ x_2, \ldots \}$

$x_5 \leftrightarrow \{ x_0, x_4, x_5, \ldots \}$

... $\leftrightarrow \{ \ldots \}$
\[
\begin{array}{cccccc}
  x_0 & x_1 & x_2 & x_3 & x_4 & x_5 & \ldots \\
  \{ x_0, \} & x_2, & x_4, & \ldots & \{ \} \\
  x_1 & \{ \} & x_3, & x_5, & \ldots & \{ \} \\
  x_2 & \{ x_0, \} & x_2, & x_5, & \ldots & \{ \} \\
  x_3 & \{ \} & x_1, & x_4, & \ldots & \{ \} \\
  x_4 & \{ \} & x_2, & \ldots & \{ \} \\
  x_5 & \{ x_0, \} & \ldots & x_4, & x_5, & \ldots & \{ \} \\
  \ldots & \{ \} & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \{ \} \\
\end{array}
\]
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>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_0$</td>
<td>{ $x_0, x_2, x_4, \ldots$ }</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_1$</td>
<td>{                      }</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_2$</td>
<td>{ $x_0, x_2, x_5, \ldots$ }</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_3$</td>
<td>{ $x_1, x_4, \ldots$ }</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_4$</td>
<td>{ $x_2, \ldots$ }</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_5$</td>
<td>{ $x_0, x_4, x_5, \ldots$ }</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>{ $\ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots$ }</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

“Flip” which elements are included.
Which element is paired with this set?
Which element is paired with this set?

It differs from every set in at least one position!

```
x0  x1  x2  x3  x4  x5  ...
\hline
x0  | x0, | x2, | x4, | ... |
------
x1  |     | x3, | x5, | ... |
------
x2  | x0, | x2, | x5, | ... |
------
x3  | x1, |     | x4, | ... |
------
x4  |     | x2, |     | ... |
------
x5  | x0, | x4, | x5, | ... |
------
...  |     |     |     |     |
------
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 Cantor’s diagonalization proof 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

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

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?
All Turing machines, listed in some order
All descriptions of Turing machines, listed in the same order.
<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>$\cdots$</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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_2$</td>
<td></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>
<td></td>
</tr>
<tr>
<td>$M_4$</td>
<td></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>
<td></td>
</tr>
<tr>
<td>$\cdots$</td>
<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>
<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>$M_2$</td>
<td></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>
<td></td>
</tr>
<tr>
<td>$M_4$</td>
<td></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>
<td></td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The table shows the similarity scores with 'Acc' (Accurate) or 'No' (Not Accurate) for each pair of models.
<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>...</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>...</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></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>
<td>...</td>
</tr>
<tr>
<td>$M_5$</td>
<td></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>
<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></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>
<td></td>
</tr>
<tr>
<td>$\cdots$</td>
<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>$\cdots$</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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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>( \ldots )</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>( \ldots )</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>( \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></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>$\cdots$</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></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>$M_0$</td>
<td>$M_1$</td>
<td>$M_2$</td>
<td>$M_3$</td>
<td>$M_4$</td>
<td>$M_5$</td>
<td>\ldots</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>\ldots</td>
<td></td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>\ldots</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>No</td>
<td>\ldots</td>
<td></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></td>
</tr>
<tr>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>\ldots</td>
<td></td>
</tr>
</tbody>
</table>
Flip all “accept”s to “no”s and vice versa.
<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>$\ldots$</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>
</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>No</td>
<td>No</td>
<td>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

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>$\cdots$</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>No</td>
<td>Acc</td>
<td>No</td>
<td>Acc</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>(\ldots)</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>(\ldots)</td>
<td></td>
</tr>
<tr>
<td>(M_0)</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>Acc</td>
<td>(\ldots)</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>(\ldots)</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>(\ldots)</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>(\ldots)</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>(\ldots)</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>(\ldots)</td>
<td></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></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>$\cdots$</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>Acc</td>
<td>No</td>
<td>Acc</td>
<td>$\cdots$</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></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>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td></td>
</tr>
<tr>
<td>$\langle M_6 \rangle$</td>
<td>$\langle M_7 \rangle$</td>
<td>$\langle M_8 \rangle$</td>
<td>$\langle M_9 \rangle$</td>
<td>$\langle M_{10} \rangle$</td>
<td>$\langle M_{11} \rangle$</td>
<td>$\cdots$</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>$M_0$</td>
<td>$M_1$</td>
<td>$M_2$</td>
<td>$M_3$</td>
<td>$M_4$</td>
<td>$M_5$</td>
<td>$\cdots$</td>
<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>$\cdots$</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>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td>$\cdots$</td>
<td></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></td>
</tr>
</tbody>
</table>
\[
\begin{array}{cccccc}
\langle M_0 \rangle & \langle M_1 \rangle & \langle M_2 \rangle & \langle M_3 \rangle & \langle M_4 \rangle & \langle M_5 \rangle & \cdots \\
M_0 & \text{Acc} & \text{No} & \text{No} & \text{Acc} & \text{Acc} & \text{No} & \cdots \\
M_1 & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \cdots \\
M_2 & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \cdots \\
M_3 & \text{No} & \text{Acc} & \text{Acc} & \text{No} & \text{Acc} & \text{Acc} & \cdots \\
M_4 & \text{Acc} & \text{No} & \text{Acc} & \text{No} & \text{Acc} & \text{No} & \cdots \\
M_5 & \text{No} & \text{No} & \text{Acc} & \text{Acc} & \text{No} & \text{No} & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\text{No} & \text{No} & \text{No} & \text{Acc} & \text{No} & \text{Acc} & \cdots \\
\end{array}
\]

No TM has this behavior!
<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>$\ldots$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$M_0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$M_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$M_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$M_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$M_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>No</strong></td>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td><strong>No</strong></td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$M_5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td><strong>No</strong></td>
<td><strong>No</strong></td>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td><strong>No</strong></td>
<td><strong>Acc</strong></td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

“The language of all TMs that do not accept their descriptions”
\[
\begin{array}{cccccccc}
\langle M_0 \rangle & \langle M_1 \rangle & \langle M_2 \rangle & \langle M_3 \rangle & \langle M_4 \rangle & \langle M_5 \rangle & \cdots \\
M_0 & \text{Acc} & \text{No} & \text{No} & \text{Acc} & \text{Acc} & \text{No} & \cdots \\
M_1 & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \cdots \\
M_2 & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \text{Acc} & \cdots \\
M_3 & \text{No} & \text{Acc} & \text{Acc} & \text{No} & \text{Acc} & \text{Acc} & \cdots \\
M_4 & \text{Acc} & \text{No} & \text{Acc} & \text{No} & \text{Acc} & \text{No} & \cdots \\
M_5 & \text{No} & \text{No} & \text{Acc} & \text{Acc} & \text{No} & \text{No} & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
\text{No} & \text{No} & \text{No} & \text{Acc} & \text{No} & \text{Acc} & \cdots \\
\end{array}
\]

\{ \langle M \rangle \mid M \text{ is a TM that does not accept } \langle M \rangle \}
The *diagonalization language*, which we denote $L_D$, is defined as

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

That is, $L_D$ is the set of descriptions of Turing machines that do not accept themselves.

We constructed this language to be different from the language of every TM, so $L_D \notin \text{RE}$. Let’s prove this!
\[ L_D = \{ \langle M \rangle \mid M \text{ is a TM and } M \text{ does not accept } \langle M \rangle \} \]

**THEOREM.** \( L_D \not\in \text{RE}. \)

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

What happens when we run \( R \) on \( \langle R \rangle \)? We know that \( R \) accepts \( \langle R \rangle \) if and only if \( \langle R \rangle \in L(R). \)

Since \( L(R) = L_D, \) this is equivalent to saying:

\( R \) 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 \( \langle R \rangle. \) Therefore,

\( R \) 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 \( L_D \not\in \text{RE}. \) ■
Regular languages

Context-free languages

$R$

$RE$

$\text{HALT}_{TM}$

$A_{TM}$

$\mathbb{L}_{D}$

All languages
Acknowledgments

This lecture incorporates material from:

W. Daniel Hillis
Nancy Ide
Keith Schwarz
Michael Sipser