**Undecidable** - No algorithm

Problem \( X \) is undecidable iff

- there is no algorithm to solve arbitrary instances of \( X \) in finite time

Typically, undecidable problems ask questions about code (Turing machines, Python, C, ...)

Some questions about code are decidable


---

**Canonical: Halting problem:**

- Given a program \( <M> \)
- an input string \( x \)
- Does \( M \) halt when given input \( x \)?

**Self-Halt:** Given program \( <M> \)

- Does \( M \) halt when given \( <M> \) as input?

Suppose Bob decides **SELF-HALT**.

\[ \begin{align*}
\text{Accept} (\text{Bob}) &= \text{SELF-HALT} \\
\text{Hang} (\text{Bob}) &= \emptyset
\end{align*} \]

\[
\begin{align*}
\text{Alice} (w) : & \quad \text{If Bob} (w) \text{ accepts } \\
& \quad \text{else return True} \\
\text{Accept} (\text{Alice}) = \text{Reject} (\text{Bob})
\end{align*}
\]

\[
\begin{align*}
\text{Alice accepts} (\text{Alice}) & \implies \text{Bob accepts} (\text{Alice}) \\
& \implies \text{Alice hangs on} (\text{Alice}) \\
& \implies \text{Bob rejects} (\text{Alice}) \\
& \implies \text{Alice accepts} (\text{Alice}) \\
& \implies \text{Bob doesn't exist!}
\end{align*}
\]

by def. SELF-HALT
by def. Alice
by def. SELF-HALT
**Thm:** 

$\text{HALT is undecidable}$

**Proof:** reduction from SelfHalt.

To prove problem $X$ is undecidable, describe a reduction from any undecidable problem to problem $X$.

**NeverHalt:** Given $\langle M \rangle$, does $M$ always co-loop?

**Suppose Bunny decides NeverHalt**

Build an algorithm for Halt:

$$\text{HaltDecider}(\langle M \rangle, w):$$

```
Write following code:

Meow(x):
    returns M(w)

if Bunny(\langle Meow \rangle) then return False
else return True
```

**Reduction from Halt**

Suppose $M$ halts on $w$:

$\Rightarrow$ Meow halts on every input
$\Rightarrow$ Bunny rejects $\langle\text{Meow}\rangle$
$\Rightarrow$ HaltDecider accepts $\langle M \rangle, w$

Suppose $M$ loops on $w$:

$\Rightarrow$ Meow loops on every input
$\Rightarrow$ Bunny accepts $\langle\text{Meow}\rangle$
$\Rightarrow$ HaltDecider rejects $\langle M \rangle, w$

Contradiction! Bunny doesn't exist!
Rice's Theorem:

Given \(<M>\), does \(M\) accept _____?

\[ \text{Accept}(M) = \{ w \mid M \text{ accepts } w \} \]

Let \(L\) be any family of languages such that:
- There is a program \(Y\) s.t. \(\text{Accept}(Y) \in L\)
- There is a program \(N\) s.t. \(\text{Accept}(N) \notin L\)

Deciding, given \(<M>\), if \(\text{Accept}(M) \in L\) is impossible.

Does \(M\) accept \(\varepsilon\)?
- \(L = \text{languages contain } \varepsilon\)
- \(Y = \text{accept all strings}\)
- \(N = \text{accept nothing}\)

Rice's Theorem \(\checkmark\)

Does \(M\) accept all palindromes with length \(2^n\)?
- \(L = \text{lang containing all palindromes length } 2^n\)
- \(Y = \text{accept } \varepsilon^*\)
- \(N = \text{accept } \emptyset\)

Does \(M\) accept a non-regular language?
- \(Y = \text{accepts } \varepsilon 0^n 1^n 0^n \text{ and nothing else}\)
- \(N = \text{return True}\)