Rice’s Theorem
We’ve seen that analyzing programs is very hard. But can we more easily tell when some “program analysis” problem is undecidable?

**Problem 1**  Undecidable

\{ (M, w) \mid M \text{ is a TM that on input } w, \text{ tries to move its head past the left end of the input} \}

**Problem 2**  Decidable

\{ (M, w) \mid M \text{ is a TM that on input } w, \text{ moves its head left at least once, at some point} \}
Problem 1  Undecidable

$L' = \{ (M, w) \mid M \text{ is a TM that on input } w, \text{ tries to move its head past the left end of the input } \}$

Proof:  Reduce $A_{TM}$ to $L'$

On input $(M, w)$, make a TM $N$ that shifts $w$ over one cell, marks a special symbol $\$ on the leftmost cell, then simulates $M(w)$ on the tape.
If $M$’s head moves to the cell with $\$ but has not yet accepted, $N$ moves the head back to the right.
If $M$ accepts, $N$ tries to move its head past the $\$. 

$(M, w)$ is in $A_{TM}$ if and only if $(N, w)$ is in $L'$
Problem 2  Decidable

\{(M, w) | M is a TM that on input w, moves its head left at least once, at some point\}

On input \((M, w)\), run \(M\) on \(w\) for \(|Q| + |w| + 1\) steps,
where \(|Q|\) = number of states of \(M\).

Accept  If \(M\)’s head moved left at all
Reject   Otherwise

(Why does this work?)
Problem 3

REVERSE = \{ M \mid M \text{ is a TM with the property: for all } w, M(w) \text{ accepts } \Leftrightarrow M(w^R) \text{ accepts} \}.

Decidable or not?

REVERSE is undecidable.
Rice’s Theorem

Let $P : \{\text{Turing Machines}\} \to \{0, 1\}$.
(Think of 0=false, 1=true) Suppose $P$ satisfies:

1. (Nontrivial) There are TMs $M_{\text{YES}}$ and $M_{\text{NO}}$ where $P(M_{\text{YES}}) = 1$ and $P(M_{\text{NO}}) = 0$

2. (Semantic) For all TMs $M_1$ and $M_2$, If $L(M_1) = L(M_2)$ then $P(M_1) = P(M_2)$

Then, $L = \{M \mid P(M) = 1\}$ is undecidable.

A Huge Hammer for Undecidability!
Some Examples and Non-Examples

Semantic Properties $P(M)$
- $M$ accepts 0
- for all $w$, $M(w)$ accepts iff $M(w^R)$ accepts
- $L(M) = \{0\}$
- $L(M)$ is empty
- $L(M) = \Sigma^*$
- $M$ accepts 154 strings

Not Semantic!
- $M$ halts and rejects 0
- $M$ tries to move its head off the left end of the tape, on input 0
- $M$ never moves its head left on input 0
- $M$ has exactly 154 states
- $M$ halts on all inputs

$L = \{M \mid P(M) \text{ is true}\}$ is undecidable

There are $M_1$ and $M_2$ such that $L(M_1) = L(M_2)$ and $P(M_1) \neq P(M_2)$
Rice’s Theorem: If $P$ is nontrivial and semantic, then $L = \{M \mid P(M) = 1\}$ is undecidable.

Proof: Either reduce $A_{TM}$ or $\neg A_{TM}$ to the language $L$

Define $M_{\emptyset}$ to be a TM such that $L(M_{\emptyset}) = \emptyset$

Case 1: $P(M_{\emptyset}) = 0$

Since $P$ is nontrivial, there’s $M_{\text{YES}}$ such that $P(M_{\text{YES}}) = 1$

Reduction from $A_{TM}$ to $L$ On input $(M,w)$, output: “$M_w(x) := \text{If } ((M \text{ accepts } w) \& (M_{\text{YES}} \text{ accepts } x)) \text{ then ACCEPT, else REJECT}$”

If $M$ accepts $w$, then $L(M_w) = L(M_{\text{YES}})$

Since $P(M_{\text{YES}}) = 1$, we have $P(M_w) = 1$ and $M_w \in L$

If $M$ does not accept $w$, then $L(M_w) = L(M_{\emptyset}) = \emptyset$

Since $P(M_{\emptyset}) = 0$, we have $M_w \notin L$
Rice’s Theorem: If $P$ is nontrivial and semantic, then $L = \{M \mid P(M) = 1\}$ is undecidable.

**Proof:** Either reduce $A_{TM}$ or $\neg A_{TM}$ to the language $L$

Define $M_{\emptyset}$ to be a TM such that $L(M_{\emptyset}) = \emptyset$

Case 2: $P(M_{\emptyset}) = 1$

Since $P$ is nontrivial, there’s $M_{NO}$ such that $P(M_{NO}) = 0$

Reduction from $\neg A_{TM}$ to $L$  On input $(M,w)$, output:

“$M_{w}(x) := \text{If } ((M \text{ accepts } w) \& (M_{NO} \text{ accepts } x)) \text{ then ACCEPT, else REJECT}”$

If $M$ does not accept $w$, then $L(M_{w}) = L(M_{\emptyset}) = \emptyset$ Since $P(M_{\emptyset}) = 1$, we have $M_{w} \in L$

If $M$ accepts $w$, then $L(M_{w}) = L(M_{NO})$

Since $P(M_{NO}) = 0$, we have $M_{w} \notin L$
The Regularity Problem for Turing Machines

\[ \text{REGULAR}_{TM} = \{ M \mid M \text{ is a TM and } L(M) \text{ is regular} \} \]

Given a program, is it equivalent to some DFA?

**Theorem:** \( \text{REGULAR}_{TM} \) is not recognizable

**Proof:** Use Rice’s Theorem!

\[ P(M) := \text{“} L(M) \text{ is regular”} \] is nontrivial:
- there’s an \( M_\emptyset \) such that \( L(M_\emptyset) = \emptyset \): \( P(M_\emptyset) = 1 \)
- there’s an \( M' \) deciding \( \{0^n1^n \mid n \geq 0\} \): \( P(M') = 0 \)

\( P \) is also semantic:
If \( L(M) = L(M') \) then \( L(M) \) is regular iff \( L(M') \) is regular, so \( P(M) = 1 \) iff \( P(M') = 1 \), so \( P(M) = P(M') \)

By Rice’s Thm, we have \( \neg A_{TM} \leq_m \text{REGULAR}_{TM} \)
Parting thoughts:
Properties about computations are hard to compute for a very wide class of computations.