CS 154

More on Reductions,
Rice’s Theorem
Reducing One Problem to Another

$f : \Sigma^* \rightarrow \Sigma^*$ is a computable function if there is a Turing machine $M$ that halts with just $f(w)$ written on its tape, for every input $w$.

A language $A$ is *mapping reducible* to language $B$, written as $A \leq_m B$, if there is a computable $f : \Sigma^* \rightarrow \Sigma^*$ such that for every $w$,

$$w \in A \iff f(w) \in B$$

$f$ is called a mapping reduction (or many-one reduction) from $A$ to $B$. 
Let \( f : \Sigma^* \to \Sigma^* \) be a computable function such that \( w \in A \iff f(w) \in B \).

Say: “\( A \) is mapping reducible to \( B \)”

Write: \( A \leq_m B \)
Examples

\[ A_{DFA} = \{ (D, w) \mid D \text{ encodes a DFA over some } \Sigma, \]
\[ \text{and } D \text{ accepts } w \in \Sigma^* \} \]
\[ A_{NFA} = \{ (N, w) \mid N \text{ encodes an NFA, } D \text{ accepts } w \} \]

Theorem: \( A_{DFA} \leq_m A_{NFA} \)

Every DFA can be trivially written as an NFA.
So one mapping reduction \( f \) from \( A_{DFA} \) to \( A_{NFA} \) is:
\[ f(D,w) := \text{Construct NFA } N \text{ which is equivalent to } D \]
\[ \text{Output } (N,w) \]

Theorem: \( A_{NFA} \leq_m A_{DFA} \)

\[ f(N,w) := \text{Use the subset construction to convert } \]
\[ \text{NFA } N \text{ into an equivalent DFA } D. \text{ Output } (D,w) \]
Theorem: If $A \leq_m B$ and $B$ is decidable, then $A$ is decidable

Corollary: If $A \leq_m B$ and $A$ is undecidable, then $B$ is undecidable

Theorem: If $A \leq_m B$ and $B$ is recognizable, then $A$ is recognizable

Corollary: If $A \leq_m B$ and $A$ is unrecognizable, then $B$ is unrecognizable
Theorem: \( A_{TM} \leq_m \text{HALT}_{TM} \)

Define
\[
f(z) := \text{Decode } z \text{ into a pair } (M, w)
\]
Construct \( M' \) with the specification:
“\( M'(w) = \text{Simulate } M \text{ on } w. \)
if \( M(w) \) accepts then \textit{accept}
else \textit{loop forever}”

Output \( (M', w) \)

We have \( z \in A_{TM} \iff (M', w) \in \text{HALT}_{TM} \)
Theorem: $A_{TM} \leq_m HALT_{TM}$

Corollary: $\neg A_{TM} \leq_m \neg HALT_{TM}$

Corollary: $\neg HALT_{TM}$ is unrecognizable!

Proof: If $\neg HALT_{TM}$ were recognizable, then $\neg A_{TM}$ would be recognizable...
Theorem: $\text{HALT}_{\text{TM}} \leq_m A_{\text{TM}}$

Proof: Define the computable function:

$$f(z) := \text{Decode } z \text{ into a pair } (M, w)$$

Construct $M'$ with the specification:

"$M'(w) = \text{Simulate } M \text{ on } w$.
\hspace{1cm} \text{If } M(w) \text{ halts then accept}
\hspace{1cm} \text{else loop forever}"

Output $(M', w)$

Observe $(M, w) \in \text{HALT}_{\text{TM}} \iff (M', w) \in A_{\text{TM}}$
Corollary: $\text{HALT}_{\text{TM}} \equiv_m A_{\text{TM}}$

Yo, T.M.! I can give you the magical power to either compute the halting problem, or the acceptance problem. Which do you want?

Wow, hm, so hard to choose...

I can't decide!
The Emptiness Problem for TMs

$$\text{EMPTY}_{\text{TM}} = \{ M \mid M \text{ is a TM such that } L(M) = \emptyset \}$$

Given a program, does it reject or loop on every input?

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

Proof: Show that \( \neg A_{\text{TM}} \leq_m \text{EMPTY}_{\text{TM}} \)

\[ f(z) := \text{Decode } z \text{ into a pair } (M, w). \]

Output a TM \( M' \) with the behavior:

“\( M'(x) := \text{if } (x = w) \text{ then output answer of } M(w), \text{ else reject} \)”

\[ z \notin A_{\text{TM}} \iff M \text{ doesn’t accept } w \]

\[ \iff L(M') = \emptyset \iff M' \in \text{EMPTY}_{\text{TM}} \]

\[ \iff f(z) \in \text{EMPTY}_{\text{TM}} \]
The Emptiness Problem for Other Stuff

$\text{EMPTY}_{\text{DFA}} = \{ M \mid M \text{ is a DFA such that } L(M) = \emptyset \}$

Given a DFA, does it reject every input?

Theorem: $\text{EMPTY}_{\text{DFA}}$ is decidable

Why?

$\text{EMPTY}_{\text{NFA}} = \{ M \mid M \text{ is a NFA such that } L(M) = \emptyset \}$

$\text{EMPTY}_{\text{REX}} = \{ R \mid M \text{ is a regexp such that } L(M) = \emptyset \}$
The Equivalence Problem

\[ \text{EQ}_{\text{TM}} = \{(M, N) \mid M, N \text{ are TMs and } L(M) = L(N)\} \]

*Do two programs compute the same function?*

Theorem: \( \text{EQ}_{\text{TM}} \) is *unrecognizable*

Proof: Reduce \( \text{EMPTY}_{\text{TM}} \) to \( \text{EQ}_{\text{TM}} \)

Let \( M_\emptyset \) be a TM that always loops forever, so \( L(M_\emptyset) = \emptyset \)

Define \( f(M) := (M, M_\emptyset) \)

\[ M \in \text{EMPTY}_{\text{TM}} \iff L(M) = L(M_\emptyset) \iff (M, M_\emptyset) \in \text{EQ}_{\text{TM}} \]
Moral:
Analyzing Programs is Really, Really Hard.

How can we more easily tell when some “program analysis” problem is undecidable?
Problem 1  Undecidable
{ (M, w) | M is a TM that on input w, tries to move its head past the left end of the input }

Problem 2  Decidable
{ (M, w) | M is a TM that on input w, moves its head left at least once, at some point }
Problem 1    Undecidable

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

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

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:} \\
\text{for all } w, M(w) \text{ accepts } \iff M(w^R) \text{ accepts} \}.

Decidable or not?

REVERSE is undecidable.
Rice’s Theorem

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

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

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 \( M_{YES} \) such that \( P(M_{YES}) = 1 \).

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

\[
M_w(x) := \text{If } ((M \text{ accepts } w) \& (M_{YES} \text{ accepts } x)) \text{ then ACCEPT, else REJECT}
\]

If \( M \) accepts \( w \), then \( L(M_w) = L(M_{YES}) \)
Since \( P(M_{YES}) = 1 \), 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 $M_{\neg\neg\neg\neg}$ such that $P(M_{\neg\neg\neg\neg}) = 0$

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

"$M_w(x) := \text{If } ((M \text{ accepts } w) \& (M_{\neg\neg\neg\neg} \text{ accepts } x)) \text{ then ACCEPT, else REJECT}"$

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 $L(M_w) = L(M_{\neg\neg\neg\neg})$

Since $P(M_{\neg\neg\neg\neg}) = 0$, we have $M_w \notin L"
The Regularity Problem for Turing Machines

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

*Given a program, is it equivalent to some DFA?*

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

**Proof:** Use Rice’s Theorem!

\( P(M) := \text{“}L(M) \text{ is regular”} \) is nontrivial:
- There’s an \( M_\emptyset \) which never halts: \( P(M_\emptyset) = 1 \)
- 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 \( \neg A_{\text{TM}} \leq_m \text{REGULAR}_{\text{TM}} \)
Recognizability via Logic

Def. A decidable predicate \( R(x,y) \) is a proposition about the input strings \( x \) and \( y \), such that some TM \( M \) implements \( R \). That is,

- for all \( x, y \), \( R(x,y) \) is \( \text{TRUE} \) \( \Rightarrow \) \( M(x,y) \) accepts
- \( R(x,y) \) is \( \text{FALSE} \) \( \Rightarrow \) \( M(x,y) \) rejects

Can think of \( R \) as a function from \( \Sigma^* \times \Sigma^* \rightarrow \{\text{T,F}\} \)

EXAMPLES: \( R(x,y) = \) “\( xy \) has at most 100 zeroes”
\( R(N,y) = \) “TM \( N \) halts on \( y \) in at most 99 steps”
Theorem: A language $A$ is *recognizable* if and only if there is a decidable predicate $R(x, y)$ such that:

$$A = \{ x \mid \exists y \ R(x, y) \}$$

Proof: (1) If $A = \{ x \mid \exists y \ R(x,y) \}$ then $A$ is recognizable

Define the TM $M(x)$: For all finite-length strings $y$, If $R(x,y)$ is true, accept.

Then, $M$ accepts exactly those $x$ s.t. $\exists y \ R(x,y)$ is true

(2) If $A$ is recognizable, then $A = \{ x \mid \exists y \ R(x,y) \}$

Suppose TM $M$ recognizes $A$.

Let $R(x,y)$ be TRUE iff $M$ accepts $x$ in $|y|$ steps.

Then, $M$ accepts $x$ $\iff \exists y \ R(x,y)$