CS 154

**Rice's Theorem,
the Recursion Theorem,
and the Fixed-Point Theorem**

Next Tuesday (2/14)

Midterm in class

On Thursday: instead of a new homework, you'll get some sample midterm questions

Problem 1

REVERSE = { M | M is a TM with the property: for all w, M(w) accepts iff M(w^R) accepts}.

REVERSE is undecidable.

x → M_w

If x = 01 accept

If x = 10 run M(w) →

Given a machine D for deciding the language REVERSE, we show how to decide A_{TM}

N → D →

N accepts w^R if it accepts w

Problem 2.1 **UNDECIDABLE**

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

Problem 2.2 **DECIDABLE**

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

Problem 2.1 **UNDECIDABLE**

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

Proof: Reduce from A_{TM} to the above language

On input (M,w), make a TM N that marks the leftmost tape cell, shifts input w over one square, then simulates M(w). If M moves to the marked cell, N moves the head back to the right. If M accepts, N tries to moves its head past the left end of the tape.

(M,w) is in A_{TM} if and only if (N,w) has the property

Problem 2.2 **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 the machine for $|Q_M| + |w| + 1$ steps:

Accept	If M's head moved left at all
Reject	Otherwise

(Why does this work??)

Problem 3

Let L be a language over Turing machines. Assume that L satisfies the following properties:

1. (Semantic) For any TMs M_1 and M_2, where $L(M_1) = L(M_2)$, $M_1 \in L$ if and only if $M_2 \in L$
2. (Nontrivial) There are TMs M_{YES} and M_{NO}, where $M_{YES} \in L$ and $M_{NO} \notin L$

Prove that L is undecidable

Examples and Non-Examples

Semantic Properties P	Not Semantic!
• M accepts ε • L(M) = {0} • L(M) is empty • L(M) is regular • M accepts exactly 154 strings	• M halts and rejects ε • M tries to move its head off the left end of the tape, on input ε • M never moves its head left on input ε • M has exactly 154 states • M halts on all inputs

L = {M | P(M) is true} is undecidable

Theorem: Any semantic nontrivial L over Turing machines is undecidable

Proof:
We'll reduce from A_{TM} to the language L

Define $M_\varnothing$ to be a TM that never halts

Assume, WLOG, that $M_\varnothing \notin L$

Let $M_{YES} \in L$ (such M_{YES} exists, by assumption)

Reduction from A_{TM}: On input (M,w):

Output "$M_w(x)$:= If (M accepts w) & (M_{YES} accepts x), ACCEPT"

If M accepts w, then $L(M_w) = L(M_{YES})$
Since $M_{YES} \in L$, we have $M_w \in L$

If M does not accept w, then $L(M_w) = L(M_\varnothing) = \varnothing$
Since $M_\varnothing \notin L$, we have $M_w \notin L$

If $M_\varnothing \in L$, then we reduce $\neg A_{TM}$ to L. Define:
$M_w(x)$:= if M accepts w and
M_{NO} accepts x, ACCEPT

Rice's Theorem

Let L be a language over Turing machines. Assume that L satisfies the following properties:

1. (Semantic) For any TMs M_1 and M_2, where $L(M_1) = L(M_2)$, $M_1 \in L$ if and only if $M_2 \in L$
2. (Nontrivial) There are TMs M_{YES} and M_{NO}, where $M_{YES} \in L$ and $M_{NO} \notin L$

Then L is undecidable

"*Every nontrivial semantic property of Turing machines is undecidable*"

Extremely Powerful Stuff

One of these is recognizable, the other one is not. **Which one is which?**

{M | L(M) contains at most 154 strings}

{M | L(M) contains at least 154 strings}

Is there a generic condition for unrecognizability?

Rice's Theorem, Part II

Let L be a language over Turing machines. Assume that L satisfies the following properties:

1. (Semantic) For any TMs M_1 and M_2, where $L(M_1) = L(M_2)$, $M_1 \in L$ if and only if $M_2 \in L$

2. (Non-monotone) There are TMs M_{YES} and M_{NO}, where $M_{YES} \in L$, $M_{NO} \notin L$, and $L(M_{YES}) \subset L(M_{NO})$

Then L is unrecognizable
"Every non-monotone semantic property of Turing machines is unrecognizable"

Idea: Give a mapping reduction from $\neg A_{TM}$ to L

Examples and Non-Examples

Monotone Properties P	Non-Monotone
• L(M) is infinite	• L(M) is finite
• $L(M) = \Sigma^*$	• L(M) = {0}
• L(M) contains at least 154 strings	• L(M) is regular
• L(M) contains 11111	• L(M) is not regular
	• L(M) contains at most 154 strings

Monotone: $\forall M_{YES}, M_{NO}$, If $M_{YES} \in L$ and $L(M_{YES}) \subset L(M_{NO})$ then $M_{NO} \in L$.

L = {M | P(M) is true} is unrecognizable

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

Output "$M_w(x)$:= Run $M_{YES}(x)$, $M_{NO}(x)$, M(w) in ||
If (M accepts w) & (M_{NO} accepts x), ACCEPT
If (M_{YES} accepts x), ACCEPT"

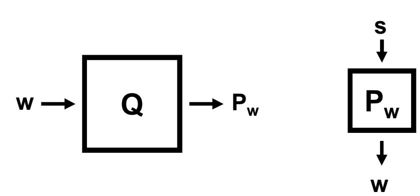
If M accepts w, then $L(M_w) = L(M_{NO})$, since $L(M_{YES}) \subset L(M_{NO})$. We have $M_w \notin L$

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

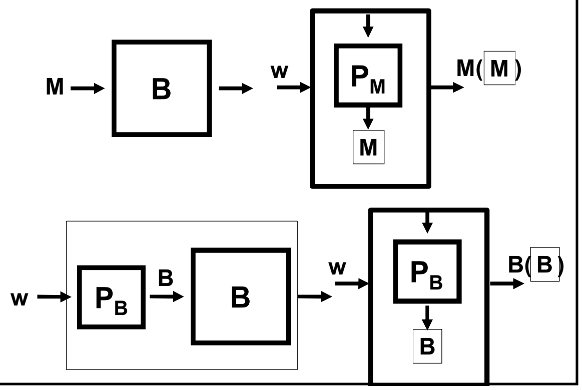
(M, w) in A_{TM} if and only if $M_w \notin L$

Self-Reference and the Recursion Theorem

Theorem: There is a computable function $q : \Sigma^* \to \Sigma^*$, where for any string w, q(w) is the *description* of a TM P_w that on any input, prints out w and then accepts



Self-Printing Turing Machine



Another Way of Looking At It

Suppose in general we want to design a program that prints its own description. **How?**

"Print this sentence."

Print two copies of the following, the second copy in quotes: **= B**

"Print two copies of the following, the second copy in quotes:" **= P_B**

The Recursion Theorem

Theorem: Let T be a Turing machine that computes a function $t : \Sigma^* \times \Sigma^* \to \Sigma^*$. There is a Turing machine R that computes a function $r : \Sigma^* \to \Sigma^*$, where for every string w,

$$r(w) = t(R, w)$$