BU CS 332 – Theory of Computation

Lecture 16:

Reductions:
- Via Computational History
- Mapping Reductions
- Post's correspondence game

Reading:

Sipser Ch. 5

Ran Canetti

November 3, 2020

Problems in language theory

A_{DFA} decidable	A_{CFG} decidable	A_{TM} **undecidable**
E_{DFA} decidable	E_{CFG} decidable	E_{TM} **undecidable**
EQ_{DFA} decidable	EQ_{CFG} ?	EQ_{TM} ?

Equality Testing for TMs

$EQ_{\text{TM}} = \{\langle M_1, M_2\rangle \mid M_1, M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$

Theorem: EQ_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for EQ_{TM}. We construct a decider for A_{TM} as follows:

On input $\langle M, w\rangle$:

1. Construct TMs M_1, M_2 as follows:

$M_1(x)$:
if $x = w$ then accept; else reject.

$M_2(x)$:
if $x = w$ run $M(w)$ if $M(w)$ accepts then accept. else reject

need to show:
1. if $\langle M, w\rangle \in A_{TM}$ then $\langle M_1, M_2\rangle \in EQ_{TM}$
2. if $\langle M, w\rangle \notin A_{TM}$ then $\langle M_1, M_2\rangle \notin EQ_{TM}$

2. Run R on input $\langle M_1, M_2\rangle$
3. If R accepts, accept. Otherwise, reject.

This is a reduction from A_{TM} to EQ_{TM}

Equality Testing for TMs

$$EQ_{\mathrm{TM}} = \{\langle M_1, M_2\rangle \mid M_1, M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem: EQ_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for EQ_{TM}. We construct a decider for A_{TM} as follows:

On input $\langle M, w\rangle$:

1. Construct TMs M_1, M_2 as follows:

   M_1 = M_2 =

2. Run R on input $\langle M_1, M_2\rangle$
3. If R accepts, accept. Otherwise, reject.

This is a reduction from A_{TM} to EQ_{TM}

Problems in language theory

A_{DFA} decidable	A_{CFG} decidable	A_{TM} undecidable
E_{DFA} decidable	E_{CFG} decidable	E_{TM} undecidable
EQ_{DFA} decidable	EQ_{CFG} ?	EQ_{TM} undecidable

TA_{TM} $\{\langle \underline{M}, \underline{w}, \underline{t} \rangle \mid M$ accepts w within t steps of computation $\}$

is TA_{TM} decidable?

decidable: Run $U(\langle M, w \rangle)$ for t steps

⏵ ⏵ Warning ⏵ ⏵

What's wrong with the following "proof"?

Bogus "Theorem": A_{TM} is not Turing-recognizable

Bogus "Proof": Suppose for contradiction that there exists a recognizer R for A_{TM}. We construct a recognizer for $\overline{A_{\mathrm{TM}}}$:

On input $\langle M, w \rangle$:
1. Run R on input $\langle M, w \rangle$
2. If R accepts, reject. Otherwise, accept.

This sure looks like a reduction from $\overline{A_{\mathrm{TM}}}$ to A_{TM}

Mapping Reductions

"many to one" reductions

Mapping Reductions: Motivation

1. How do we formalize the notion of a reduction?
2. How do we use reductions to show that languages are unrecognizable?
3. How do we protect ourselves from accidentally “proving” bogus theorems about recognizability?

Computable Functions

Definition:

A function $f: \Sigma^* \rightarrow \Sigma^*$ is computable if there is a TM M which, given as input any $w \in \Sigma^*$, halts with only $f(w)$ on its tape.

Example 1: $f(\langle x, y \rangle) = x + y$

Example 2: $f(\langle M, w \rangle) = \langle M' \rangle$ where M is a TM, w is a string, and M' is a TM that ignores its input and simulates running M on w

Mapping Reductions

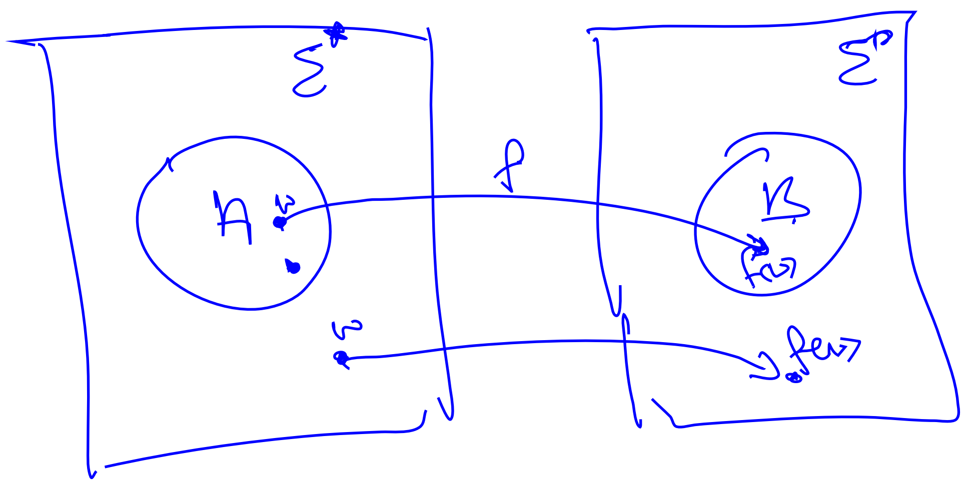
Definition:

Language A is mapping reducible to language B, written

$$A \leq_{\mathrm{m}} B$$

if there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$ such that for all strings $w \in \Sigma^*$, we have $w \in A \Leftrightarrow f(w) \in B$

$w \notin A \Rightarrow f(w) \notin B$



Decidability

Theorem: If $A \leq_{\mathrm{m}} B$ and B is decidable, then A is also decidable

Proof: Let M be a decider for B and let $f: \Sigma^* \rightarrow \Sigma^*$ be a mapping reduction from A to B. Construct a decider for A as follows:

On input w:

1. Compute $f(w)$
2. Run M on input $f(w)$
3. If M accepts, accept. Otherwise, reject.

Undecidability

Theorem: If $A \leq_{\mathrm{m}} B$ and B is decidable, then A is also decidable

Corollary: If $A \leq_{\mathrm{m}} B$ and A is undecidable, then B is also undecidable

Old Proof: Equality Testing for TMs

$$EQ_{\mathrm{TM}} = \{\langle M_1, M_2\rangle \mid M_1, M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem: EQ_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for EQ_{TM}. We construct a decider for A_{TM} as follows:

On input $\langle M, w\rangle$:

1. Construct TMs M_1, M_2 as follows:

M_1 = "On input x,
1. Ignore x
2. Run M on input w
3. If M accepts, accept. Otherwise, reject."

M_2 = "On input x,
1. Ignore x and accept"

2. Run R on input $\langle M_1, M_2\rangle$
3. If R accepts, accept. Otherwise, reject.

This is a reduction from A_{TM} to EQ_{TM}

New Proof: Equality Testing for TMs

$$EQ_{\text{TM}} = \{\langle M_1, M_2 \rangle \mid M_1, M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem: $A_{\text{TM}} \leq_{\text{m}} EQ_{\text{TM}}$ hence EQ_{TM} is undecidable

Proof: The following TM computes the reduction:

On input $\langle M, w \rangle$:

1. Construct TMs M_1, M_2 as follows:

M_1 = "On input x,
1. Ignore x
2. Run M on input w
3. If M accepts, accept.
   Otherwise, reject."

M_2 = "On input x,
1. Ignore x and accept"

2. Output $\langle M_1, M_2 \rangle$

Mapping Reductions: Recognizability

Theorem: If $A \leq_{\mathrm{m}} B$ and B is recognizable, then A is also recognizable

Proof: Let M be a recognizer for B and let $f : \Sigma^* \to \Sigma^*$ be a mapping reduction from A to B. Construct a recognizer for A as follows:

On input w:

1. Compute $f(w)$
2. Run M on input $f(w)$
3. If M accepts, accept. Otherwise, reject.

Unrecognizability

Theorem: If $A \leq_{\mathrm{m}} B$ and B is recognizable, then A is also recognizable

Corollary: If $A \leq_{\mathrm{m}} B$ and A is unrecognizable, then B is also unrecognizable

Corollary: If $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} B$, then B is unrecognizable

Consequences of $A_{\mathrm{TM}} \leq_{\mathrm{m}} EQ_{\mathrm{TM}}$

1. Since A_{TM} is undecidable, EQ_{TM} is also undecidable

2. $A_{\mathrm{TM}} \leq_{\mathrm{m}} EQ_{\mathrm{TM}}$ implies $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} \overline{EQ_{\mathrm{TM}}}$
   Since $\overline{A_{\mathrm{TM}}}$ is unrecognizable, $\overline{EQ_{\mathrm{TM}}}$ is unrecognizable

EQ_{TM} itself is also unrecognizable

$$EQ_{\mathrm{TM}} = \{\langle M_1, M_2\rangle \mid M_1, M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem: $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} EQ_{\mathrm{TM}}$ hence EQ_{TM} is unrecognizable

Proof: The following TM computes the reduction:

On input $\langle M, w\rangle$:

1. Construct TMs M_1, M_2 as follows:

M_1 = "On input x,
1. Ignore x
2. Run M on input w
3. If M accepts, accept.
   Otherwise, reject."

M_2 = "On input x,
1. Ignore x and accept"

2. Output $\langle M_1, M_2\rangle$

More on Reductions and Undecidability

Problems in language theory

A_{DFA} decidable	A_{CFG} decidable	A_{TM} undecidable
E_{DFA} decidable	E_{CFG} decidable	E_{TM} undecidable
EQ_{DFA} decidable	EQ_{CFG} ?	EQ_{TM} undecidable

Undecidable problems outside language theory

Post Correspondence Problem (PCP):

Domino: $\left[\frac{a}{ab}\right]$. Top and bottom are strings.

Input: Collection of dominos.

$$\left[\frac{aa}{aba}\right], \left[\frac{ab}{aba}\right], \left[\frac{ba}{aa}\right], \left[\frac{abab}{b}\right]$$

Match: List of some of the input dominos (repetitions allowed) where top = bottom

$$\left[\frac{ab}{aba}\right], \left[\frac{aa}{aba}\right], \left[\frac{ba}{aa}\right], \left[\frac{aa}{aba}\right], \left[\frac{abab}{b}\right]$$

Problem: Does a match exist? This is undecidable

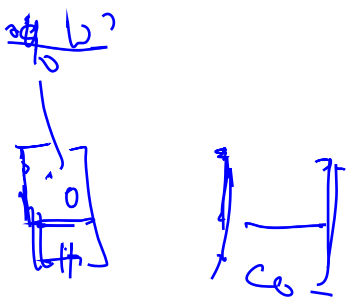
Want a mapping

$f(\langle M, w\rangle) \to$ [ ] [ ] [ ] [ ]

st if $M(w)$ accepts $\Rightarrow$ $\exists$ solution

$M(w)$ not accept/halt $\Rightarrow$ $\nexists$ solution

$\Rightarrow$ Idea: design the dominos st. any legal configuration will encode a legal sequence of conf. of $M(w)$



ALL_{CFG} is undecidable

$$ALL_{\text{CFG}} = \{\langle G \rangle \mid G \text{ is a CFG with terminal set } \Sigma \text{ and } L(G) = \Sigma^*\}$$

Theorem: $\overline{A_{\text{TM}}} \leq_{\text{m}} EQ_{\text{CFG}}$ hence EQ_{CFG} is undecidable

Proof idea: "Computation history method"

On input $\langle M, w \rangle$:

1. Construct a CFG G such that:

   $L(G) = \Sigma^* \Leftrightarrow M$ does not accept w

2. Output $\langle G \rangle$

Problems in language theory

A_{DFA} decidable	A_{CFG} decidable	A_{TM} undecidable
E_{DFA} decidable	E_{CFG} decidable	E_{TM} undecidable
EQ_{DFA} decidable	EQ_{CFG} undecidable	EQ_{TM} undecidable

An Undecidable Language

$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input } w\}$

Theorem: A_{TM} is undecidable

An Undecidable Language

$A_{\text{TM}} = \{\langle M, w \rangle \mid M \text{ is a TM that accepts input } w\}$

Theorem: A_{TM} is undecidable

Proof: Assume for the sake of contradiction that TM H decides A_{TM}:

$$H(\langle M, w \rangle) = \begin{cases} \text{accept} & \text{if } M \text{ accepts } w \\ \text{reject} & \text{if } M \text{ does not accept } w \end{cases}$$

An Undecidable Language

$A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input } w\}$

Theorem: A_{TM} is undecidable

Proof: Assume for the sake of contradiction that TM H decides A_{TM}:

$$H(\langle M, w\rangle) = \begin{cases} \text{accept} & \text{if } M \text{ accepts } w \\ \text{reject} & \text{if } M \text{ does not accept } w \end{cases}$$

Define

$$\overline{H}(\langle M, w\rangle) = \begin{cases} \text{reject} & \text{if } M \text{ accepts } w \\ \text{accept} & \text{if } M \text{ does not accept } w \end{cases}$$

Consider $H(\langle \overline{H}, w\rangle)$

An Undecidable Language

$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input } w\}$

Theorem: A_{TM} is undecidable

Proof: Assume for the sake of contradiction that TM H decides A_{TM}:

$$H(\langle M, w\rangle) = \begin{cases} \text{accept} & \text{if } M \text{ accepts } w \\ \text{reject} & \text{if } M \text{ does not accept } w \end{cases}$$

Define

$$\overline{H}(\langle M, w\rangle) = \begin{cases} \text{reject} & \text{if } M \text{ accepts } w \\ \text{accept} & \text{if } M \text{ does not accept } w \end{cases}$$

Consider $H(\langle \overline{H}, w\rangle)$: Has to run forever…

➔ H is not a decider.

An unrecognizable Language

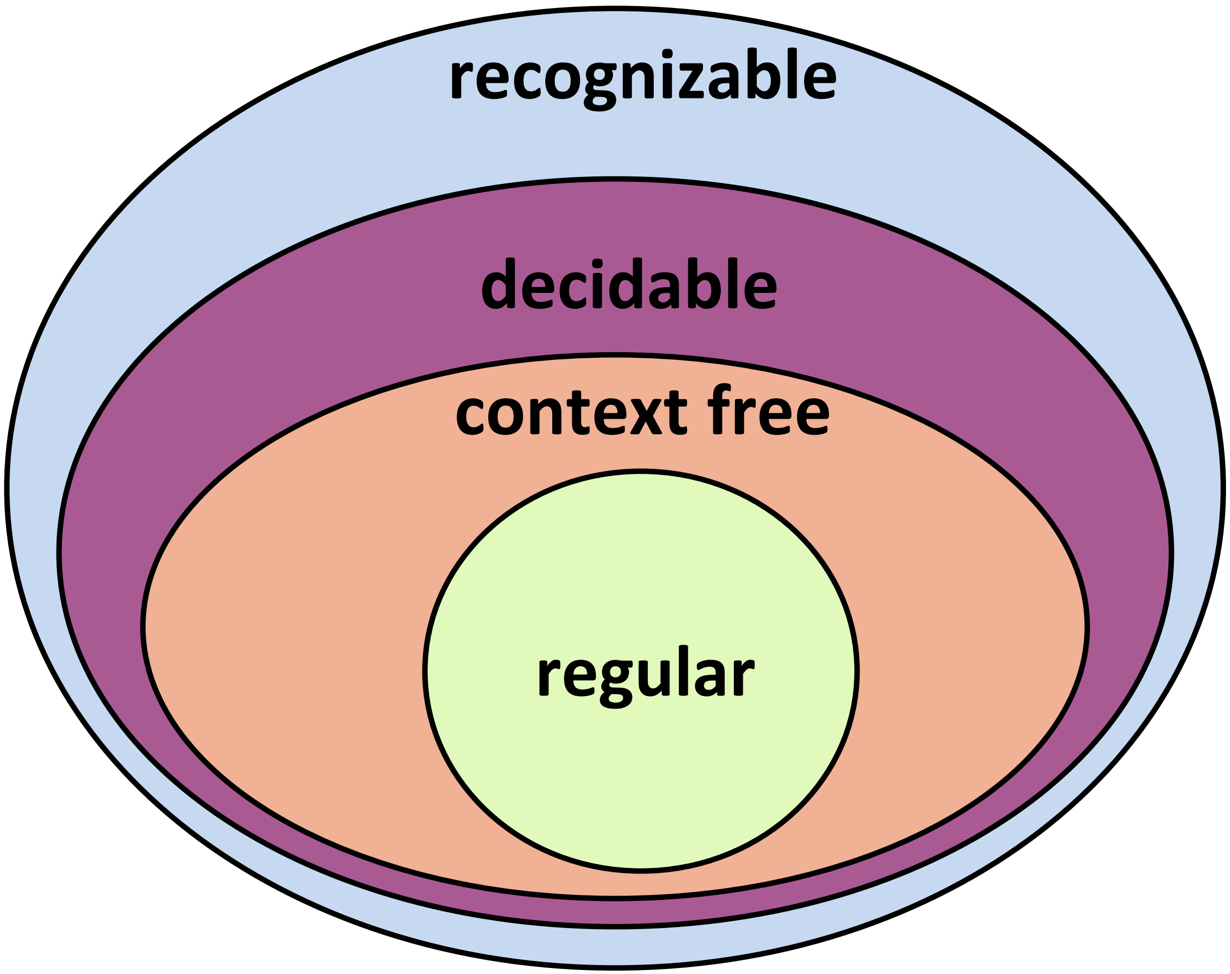
Theorem: A language L is decidable if and only if L and $\overline{L}$ are both Turing-recognizable.

($L \in \boldsymbol{R}$ if and only if both $L \in \boldsymbol{RE}$ and $\overline{L} \in \boldsymbol{RE}$)

Corollary: If L is Turing-recognizable and undecidable then $\overline{L}$ is not Turing-recognizable.

(If $L \in \boldsymbol{RE}$ and $L \notin \boldsymbol{R}$ then $\overline{L} \notin \boldsymbol{RE}$)

Classes of Languages: updated view



A specific unrecognizable Language

Theorem: A language L is decidable if and only if L and $\overline{L}$ are both Turing-recognizable.

Corollary: If L is Turing-recognizable and undecidable then $\overline{L}$ is not Turing-recognizable.

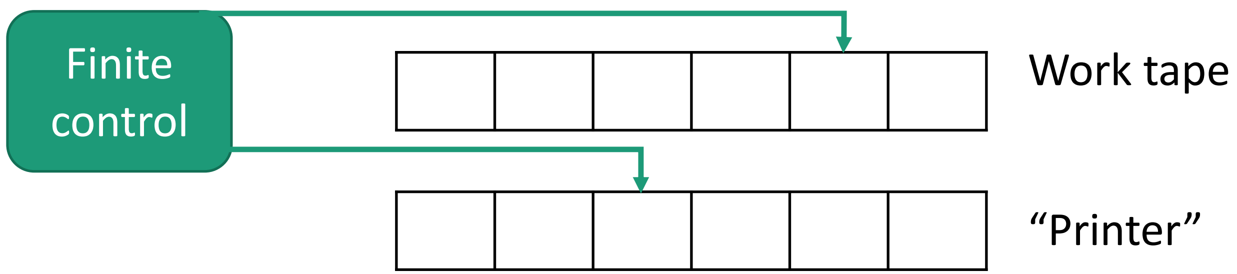
Define:

$\boldsymbol{R}$ = decidable languages

$\boldsymbol{RE}$ = Turing-recognizable languages

$\mathbf{co}\boldsymbol{RE} = \{\mathrm{L} \mid \overline{L}$ is Turing recognizable$\}$

Enumerators



- Starts with two blank tapes
- Prints strings to printer

$L(E) = \{\text{strings eventually printed by } E\}$

- May never terminate (even if language is finite)
- May print the same string many times

Enumerable = Turing-Recognizable

Theorem: A language is Turing-recognizable $\Leftrightarrow$ some enumerator enumerates it

Reductions

Reductions

A reduction from problem A to problem B is an algorithm for problem A which uses an algorithm for problem B as a subroutine

If such a reduction exists, we say "A reduces to B"

Two uses of reductions

Positive uses: If A reduces to B and B is decidable, then A is also decidable

$EQ_{\mathrm{DFA}} = \{\langle D_1, D_2\rangle \mid D_1, D_2 \text{ are DFAs and } L(D_1) = L(D_2)\}$

Theorem: EQ_{DFA} is decidable

Proof: The following TM decides EQ_{DFA}

On input $\langle D_1, D_2\rangle$, where $\langle D_1, D_2\rangle$ are DFAs:

1. Construct a DFA D that recognizes the symmetric difference $L(D_1) \triangle L(D_2)$
2. Run the decider for E_{DFA} on $\langle D\rangle$ and return its output

Two uses of reductions

Negative uses: If A reduces to B and A is undecidable, then B is also undecidable

$A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input } w\}$

Suppose H decides A_{TM}

Consider the following TM D.

On input $\langle M\rangle$ where M is a TM:

1. Run H on input $\langle M, \langle M\rangle\rangle$
2. If H accepts, accept. If H rejects, reject.

Claim: D decides

$$SA_{\text{TM}} = \{\langle M\rangle \mid M \text{ is a TM that accepts on input } \langle M\rangle\}$$

Two uses of reductions

Negative uses: If A reduces to B and A is undecidable, then B is also undecidable

Proof template:

1. Suppose to the contrary that B is decidable
2. Using B as a subroutine, construct an algorithm deciding A
3. But A is undecidable. Contradiction!

Halting Problem

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

Theorem: $HALT_{\mathrm{TM}}$ is undecidable

Proof: Suppose for contradiction that there exists a decider H for $HALT_{\mathrm{TM}}$. We construct a decider for A_{TM} as follows:

On input $\langle M, w\rangle$:

1. Run H on input $\langle M, w\rangle$
2. If H rejects, reject
3. If H accepts, simulate M on w
4. If M accepts, accept. Otherwise, reject

This is a reduction from A_{TM} to $HALT_{\mathrm{TM}}$

Empty language testing for TMs

$$E_{\text{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$$

Theorem: E_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for E_{TM}. We construct a decider for A_{TM} as follows:

On input $\langle M, w \rangle$:

1. Construct a TM M' as follows:

2. Run R on input $\langle M' \rangle$

3. If R , accept. Otherwise, reject

This is a reduction from A_{TM} to E_{TM}

Context-free language testing for TMs

$$CFL_{\mathrm{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is context} - \text{free}\}$$

Theorem: CFL_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for CFL_{TM}. We construct a decider for A_{TM} as follows:

On input $\langle M, w \rangle$:

1. Construct a TM M' as follows:

2. Run R on input $\langle M' \rangle$
3. If R accepts, accept. Otherwise, reject

This is a reduction from A_{TM} to CFL_{TM}

Context-free language testing for TMs

$$CFL_{\text{TM}} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) \text{ is context} - \text{free}\}$$

Theorem: CFL_{TM} is undecidable

Proof: Suppose for contradiction that there exists a decider R for CFL_{TM}. We construct a decider for A_{TM} as follows:

On input $\langle M, w\rangle$:

1. Construct a TM M' as follows:

   M' = "On input x,

   1. If $x \in \{0^n 1^n 2^n \mid n \geq 0\}$, accept
   2. Run TM M on input w
   3. If M accepts, accept."

2. Run R on input $\langle M'\rangle$
3. If R accepts, accept. Otherwise, reject

This is a reduction from A_{TM} to CFL_{TM}