Computational Problems, Mapping Reduction

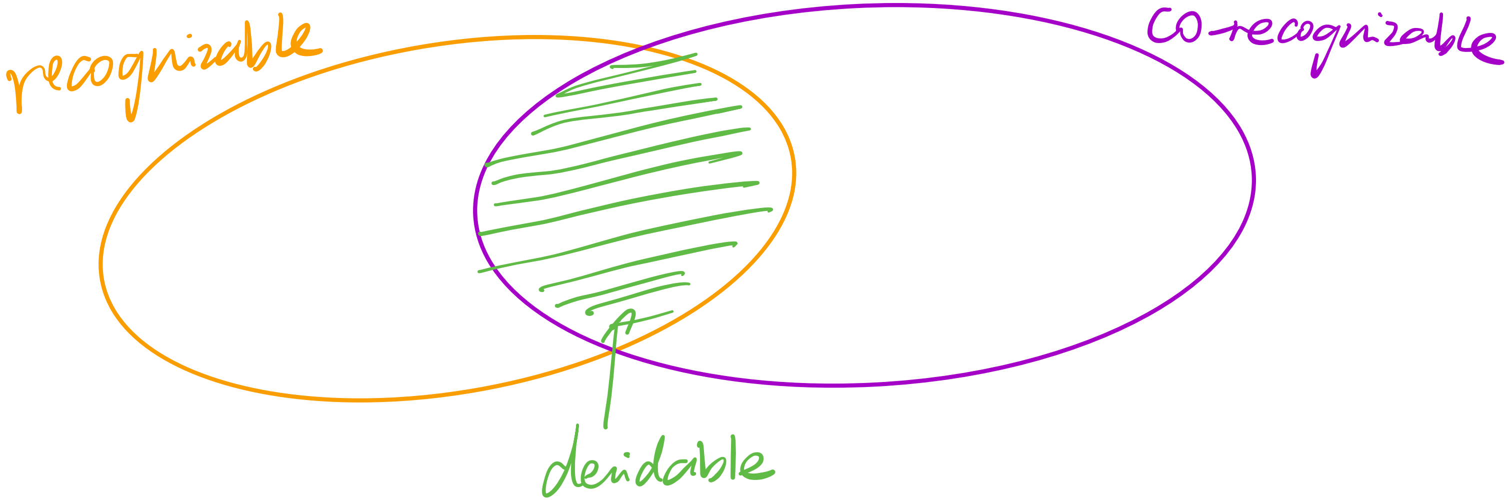
CSE 105 Week 8 Discussion

Deadlines and Logistics

- Do review quizzes on PrairieLearn
- Test 2 attempt 1 in week 10
- HW 6 due 3/13/25 at 5pm, week 10

Vocabulary check

Are all decidable languages recognizable?



Computational Problems

Computational problems

Acceptance problem	decidable	
...for DFA	A_{DFA}	$\{\langle B, w\rangle \mid B$ is a DFA that accepts input string $w\}$
...for NFA	A_{NFA}	$\{\langle B, w\rangle \mid B$ is a NFA that accepts input string $w\}$
...for regular expressions	A_{REX}	$\{\langle R, w\rangle \mid R$ is a regular expression that generates input string $w\}$
...for CFG	A_{CFG}	$\{\langle G, w\rangle \mid G$ is a context-free grammar that generates input string $w\}$
...for PDA	A_{PDA}	$\{\langle B, w\rangle \mid B$ is a PDA that accepts input string $w\}$
Language emptiness testing		
...for DFA	E_{DFA}	$\{\langle A\rangle \mid A$ is a DFA and $L(A) = \emptyset\}$
...for NFA	E_{NFA}	$\{\langle A\rangle \mid A$ is a NFA and $L(A) = \emptyset\}$
...for regular expressions	E_{REX}	$\{\langle R\rangle \mid R$ is a regular expression and $L(R) = \emptyset\}$
...for CFG	E_{CFG}	$\{\langle G\rangle \mid G$ is a context-free grammar and $L(G) = \emptyset\}$
...for PDA	E_{PDA}	$\{\langle A\rangle \mid A$ is a PDA and $L(A) = \emptyset\}$
Language equality testing		
...for DFA	EQ_{DFA}	$\{\langle A, B\rangle \mid A$ and B are DFAs and $L(A) = L(B)\}$
...for NFA	EQ_{NFA}	$\{\langle A, B\rangle \mid A$ and B are NFAs and $L(A) = L(B)\}$
...for regular expressions	EQ_{REX}	$\{\langle R, R'\rangle \mid R$ and R' are regular expressions and $L(R) = L(R')\}$
...for CFG	EQ_{CFG}	$\{\langle G, G'\rangle \mid G$ and G' are CFGs and $L(G) = L(G')\}$
...for PDA	EQ_{PDA}	$\{\langle A, B\rangle \mid A$ and B are PDAs and $L(A) = L(B)\}$

not decidable

Computational problems for Turing machines

Acceptance problem for Turing machines	A_{TM}	$\{\langle M, w\rangle \mid M \text{ is a Turing machine that accepts input string } w\}$
Language emptiness testing for Turing machines	E_{TM}	$\{\langle M\rangle \mid M \text{ is a Turing machine and } L(M) = \emptyset\}$
Language equality testing for Turing machines	EQ_{TM}	$\{\langle M_1, M_2\rangle \mid M_1 \text{ and } M_2 \text{ are Turing machines and } L(M_1) = L(M_2)\}$

What is A_{TM} ?

A. A Turing machine whose input is codes of TMs and strings.

B. A set of pairs of TMs and strings.

C. A set of strings that encode <u>TMs and strings</u>.

D. Not well defined.

E. I don't know.

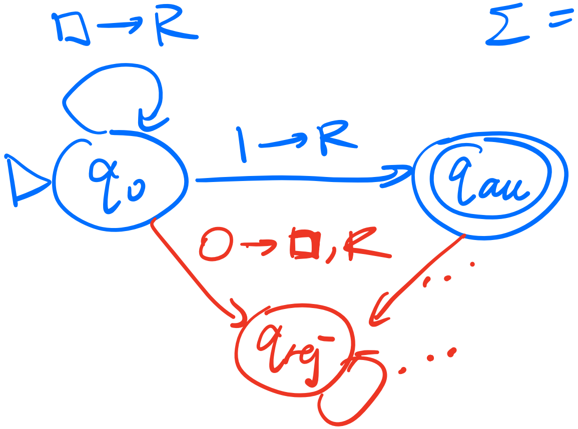
Halting problem

accept reject.

$$HALT_{TM} = \{\langle M, w\rangle \mid M \text{ is a Turing machine, } w \text{ is a string, and } \underbrace{M \text{ halts on } w}\}$$

$\Sigma = \{0,1\}$ $\Gamma = \{0, 1, \sqcup\}$

M_1:



$\langle M_1, 1\rangle \in HALT_{TM}$

$\langle M_1, 0\rangle \in HALT_{TM}$

$\langle M_1, \varepsilon\rangle \notin HALT_{TM}$

A_{TM} is recognizable but undecidable

- A_{TM} is Turing recognizable
  - We can define a Turing machine that recognizes A_{TM}
- A_{TM} is **not** Turing decidable
  - Proof by contradiction (diagonalization proof)

Define the TM N = "On input <M,w>:

1. Simulate M on w. $\rightarrow$ if M loops on w, N loops on <M,w>.
2. If M accepts, accept. If M rejects, reject."

Which of the following statements is true?

~~A.~~ N decides A_{TM}

B. N recognizes A_{TM}

~~C.~~ N always halts

D. More than one of the above.

E. I don't know

A_{TM} is recognizable but undecidable

- A_{TM} is Turing recognizable
    - We can define a Turing machine that recognizes A_{TM}
- A_{TM} is **not** Turing decidable
    - Proof by contradiction (diagonalization proof)

Proof: Suppose **towards a contradiction** that there is a Turing machine that decides A_{TM}. We call this presumed machine M_{ATM}.

Define a **new** Turing machine using the high-level description:

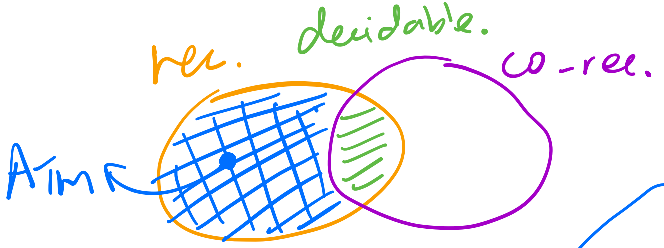
$D =$ " On input $\langle M \rangle$, where M is a Turing machine:

1. Run M_{ATM} on $\langle M, \langle M \rangle \rangle$.
2. If M_{ATM} accepts, reject; if M_{ATM} rejects, accept."

What is the result of the computation of D on $\langle D \rangle$?

A_{TM} is recognizable but undecidable

- A_{TM} is recognizable.
- A_{TM} is not decidable.
- $\overline{A_{TM}}$ is not recognizable.
- $\overline{A_{TM}}$ is not decidable.



A_{TM} not co-recognizable

Closure claims

Closure and nonclosure

Recall the definitions: A language L over an alphabet Σ is called **recognizable** if there is some Turing machine M such that $L = L(M)$. A language L over an alphabet Σ is called **co-recognizable** if its complement, defined as $\Sigma^* \setminus L = \{x \in \Sigma^* \mid x \notin L\}$, is Turing-recognizable. A language L over an alphabet Σ is called {\bf unrecognizable} if there is no Turing machine that recognizes it.

Select all and only true statements below.

$\overline{A_{TM}}$ $\overline{\overline{A_{TM}}} = A_{TM}$ not co-recognizable

- [ ] The class of co-recognizable languages is closed under complementation.

$\overline{A_{TM}}$ A_{TM} is recognizable.

- [ ] The class of unrecognizable languages is closed under complementation.
- [x] The class of decidable languages is closed under complementation.
- [ ] The class of recognizable languages is closed under complementation.

Select all possible options that apply. ? A_{TM} $\overline{A_{TM}}$ not recognizable.

Mapping Reduction

Mapping reduction

undecidable

↳ $A_{TM} \leq_m B \Rightarrow$ undecidable.

DEFINITION 5.20

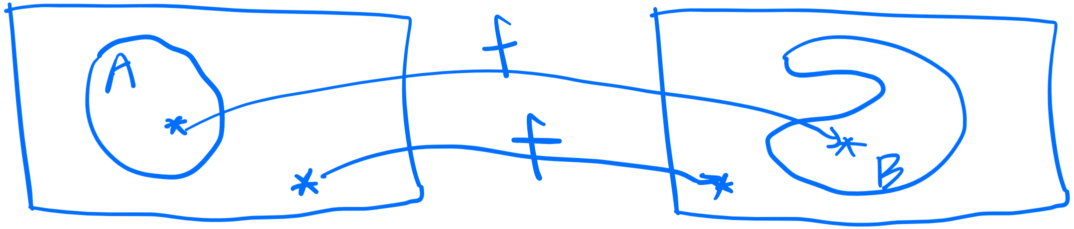
A is no harder than B

Language A is ***mapping reducible*** to language B, written $A \leq_{\mathrm{m}} B$, if there is a computable function $f: \Sigma^* \longrightarrow \Sigma^*$, where for every w,

$$w \in A \Longleftrightarrow f(w) \in B.$$

convert the question about x's membership in A into a question about $f(x)$'s membership in B.

- $f(x) \in B$, then $x \in A$
- $f(x) \notin B$, then $x \notin A$



if $x \in A$, then $f(x) \in B$

if $x \notin A$, then $f(x) \notin B$

Computable function

DEFINITION 5.17

A function $f : \Sigma^* \longrightarrow \Sigma^*$ is a ***computable function*** if some Turing machine M, on every input w, halts with just $f(w)$ on its tape.

① defined for all $x \in \Sigma^*$

② there exists an algorithm, take x as input, output $f(x)$ without entering a loop.

$$f(x) = \begin{cases} \varepsilon & \text{if } x \neq \langle N \rangle \text{ for any NFA } N \\ \langle D \rangle & \text{if } x = \langle N \rangle \text{ for some NFA } N \\ & \text{and } L(N) = L(D) \text{ where } D \text{ is DFA.} \end{cases}$$

M: "On input x:

1. if $x \neq \langle N \rangle$ for any NFA N, output ε.
2. Otherwise, $x = \langle N \rangle$. use macrostate build DFA D s.t. $L(N) = L(D)$. Output $\langle D \rangle$.

Mapping reduction

If problem X is no harder than problem Y $X \leq_m Y$

…and if Y is **decidable**

…then X must also be **decidable**

If problem X is no harder than problem Y

…and if X is **undecidable**

…then Y must also be **undecidable**

Mapping reduction practice

Fix $\Sigma = \{0, 1\}$ throughout this question.

Is each of the stated mapping reductions witnessed by the given function?

- $\Sigma^* \leq_m \Sigma^*$ is witnessed by the function $id : \Sigma^* \to \Sigma^*$ given by $id(x) = x$ for all x.

$x \in \Sigma^* \to f(x) \in \emptyset$ which is not possible!

- $\Sigma^* \leq_m \emptyset$ is witnessed by the function $id : \Sigma^* \to \Sigma^*$ given by $id(x) = x$ for all x.
- $\emptyset \leq_m \{0\}$ is witnessed by the function $id : \Sigma^* \to \Sigma^*$ given by $id(x) = x$ for all x.

1. The mapping reduction relationship is true but the given function does not witness this mapping reduction.
2. This mapping reduction is witnessed by this computable function.
3. The mapping reduction relationship is not true.

Mapping reduction practice

$\{ \langle M, w \rangle \mid M \text{ accepts } w \}$

Prove that $A_{\text{TM}} \leq_m EQ_{\text{TM}}$ → $\{ \langle M_1, M_2 \rangle \mid L(M_1) = L(M_2) \}$

exists computable function $f(x)$ s.t.

if $x \in A_{TM}$, then $f(x) \in EQ_{TM}$

if $x \notin A_{TM}$, then $f(x) \notin EQ_{TM}$

"Cannot reliably know this in finite time"

$$f(x) = \begin{cases} \langle M, M \rangle & \text{if } x = \langle M, w \rangle, M \text{ accepts } w \\ \text{constant} & \text{otherwise} \end{cases}$$

$(\text{constant} \notin EQ_{TM})$

This function is <u>not computable</u> because the condition check is undecidable

Mapping reduction practice

$x = \langle M, w \rangle \in A_{TM} \Rightarrow M$ accepts w

Prove that $A_{\text{TM}} \leq_m EQ_{\text{TM}}$

$x = \langle M, w \rangle \notin A_{TM}$

$\Rightarrow M$ rejects / loops on w

$$f(x) = \begin{cases} \langle \text{start} \rightarrow q_{\text{acc}}, M_w \rangle & \text{if } x = \langle M, w \rangle \text{ for a Turing machine } M \text{ and string } w \\ \langle \text{start} \rightarrow q_{\text{acc}}, \ \text{start} \rightarrow q_{\text{rej}} \quad q_{\text{acc}} \rangle & \text{otherwise.} \end{cases}$$

$\notin A_{TM}$

$\rightarrow \notin EQ_{TM}$

Where for each Turing machine M, we define

$M_w =$ "On input y

1. Simulate M on w.
2. If it accepts, accept.
3. If it rejects, reject."

$L(M_w) = \emptyset$

if M accepts w,
M_w accepts all strings.