Today's learning goals

• Explain what it means for a problem to be decidable.
• Justify the use of encoding.
• Give examples of decidable problems.
Computational problems

A computational problem is **decidable** iff the language encoding the problem instances is decidable.

*We won't specify the encoding.*

To prove decidable, define TM $M = \text{"On input } <\ldots>,$

1.
2. $\ldots$ "

Show (1) $L(M) = \ldots$ and (2) $M$ is a decider.
Computational problems over $\Sigma$

$A_{DFA}$ "Is a given string accepted by a given DFA?"
\{ <B,w> | B is a DFA, w in $\Sigma^*$, and w is in L(B) \}

$E_{DFA}$ "Is the language of a DFA empty?"
\{ <A> | A is a DFA over $\Sigma$, L(A) is empty \}

$EQ_{DFA}$ "Are the languages of two given DFAs equal?"
\{ <A, B> | A and B are DFA over $\Sigma$, L(A) = L(B) \}
From last week

$M_1 = \text{"On input } \langle B, w \rangle \text{, where } B \text{ is a DFA and } w \text{ is a string:

1. Simulate } B \text{ on input } w \text{ (by keeping track of states in } B, \text{ transition function of } B, \text{ etc.)
2. If the simulations ends in an accept state of } B, \text{ accept. If it ends in a non-accept state of } B, \text{ reject. \"")}$

$M_2 = \text{"On input } \langle A \rangle \text{, where } A \text{ is a DFA:

1. Mark the start state of } A.
2. Repeat until no new states get marked:
   i. Loop over states of } A \text{ and mark any unmarked state that has an incoming edge from a marked state.
3. If no final state of } A \text{ is marked, } \text{ accept; otherwise, } \text{ reject."} $
Non-emptiness?

$E'_{\text{DFA}}$ "Is the language of a DFA non-empty?"

Is this problem decidable?
A. Yes, using $M_3$ in the handout.
B. Yes, using $M_4$ in the handout.
C. Yes, both $M_3$ and $M_4$ work.
D. Yes, but not using the machines in the handout.
E. No.
Proving decidability

Claim: $\text{EQ}_{\text{DFA}}$ is decidable

Proof: WTS that $\{ <A, B> \mid A, B \text{ are DFA over } \Sigma, L(A) = L(B) \}$ is decidable. Idea: give high-level description

Step 1: construction

Will we be able to simulate $A$ and $B$?
What does set equality mean?
Can we use our previous work?
Proving decidability

**Claim:** $EQ_{DFA}$ is decidable

**Proof:** WTS that \{ <A, B> | A, B are DFA over $\Sigma$, $L(A) = L(B)$ \} is decidable. **Idea:** give high-level description

**Step 1: construction**

$$X = Y \iff (X \cap Y^c) \cup (Y \cap X^c) = \emptyset$$

Will we be able to simulate?

What does set equality mean?

Can we use our previous work?
Proving decidability

**Claim:** $E_{\text{DFA}}$ is decidable

**Proof:** WTS that \{ <A, B> | A, B are DFA over $\Sigma$, $L(A) = L(B)$ \} is decidable. **Idea:** give high-level description

**Step 1:** construction

Very high-level:

Build new DFA recognizing symmetric difference of A, B. Check if this set is empty.

\[ X = Y \iff ( (X \cap Y^c) \cup (Y \cap X^c) ) = \emptyset \]
Proving decidability

Claim: $\text{EQ}_{\text{DFA}}$ is decidable

Proof: WTS that $\{ <A, B> | A, B \text{ are DFA over } \Sigma, \text{ } L(A) = L(B) \}$ is decidable. Idea: give high-level description

Step 1: construction

Define TM $M_5$ by: $M_5 = "\text{On input } <A,B> \text{ where } A,B \text{ DFAs:}"

1. Construct a new DFA, $D$, from $A,B$ using algorithms for complementing, taking unions of regular languages such that $L(D) = \text{symmetric difference of } A \text{ and } B$.
2. Run machine $M_2$ on $<D>$.
3. If it accepts, accept; if it rejects, reject."
Proving decidability

Step 1: construction
Define TM $M_5$ by: $M_5 = \text{"On input } <A,B> \text{ where } A,B \text{ DFAs}
1. Construct a new DFA, $D$, from $A,B$ using algorithms for complementing, taking unions of regular languages such that $L(D) = \text{symmetric difference of } A \text{ and } B$.
2. Run machine $M_2$ on $<D>$.
3. If it accepts, accept; if it rejects, reject.

Step 2: correctness proof
WTS (1) $L(M_5) = EQ_{DFA}$ and (2) $M_5$ is a decider.
Computational problems

Which of the following computational problems are decidable?

A. $A_{\text{NFA}}$
B. $E_{\text{NFA}}$
C. $\text{EQ}_{\text{NFA}}$
D. All of the above
E. None of the above
Computational problems

Compare:

A. $A_{\text{REX}} = A_{\text{NFA}} = A_{\text{DFA}}, \quad E_{\text{REX}} = E_{\text{NFA}} = E_{\text{DFA}}, \quad \text{EQ}_{\text{REX}} = \text{EQ}_{\text{NFA}} = \text{EQ}_{\text{DFA}}$

B. They're all decidable, some are equal and some not.

C. They're of different types so all are different.

D. None of the above
Techniques

- **Subroutines**: can use decision procedures of decidable problems as subroutines in other algorithms
  - $A_{DFA}$
  - $E_{DFA}$
  - $EQ_{DFA}$

- **Constructions**: can use algorithms for constructions as subroutines in other algorithms
  - Converting DFA to DFA recognizing complement (or Kleene star).
  - Converting two DFA/NFA to one recognizing union (or intersection, concatenation).
  - Converting NFA to equivalent DFA.
  - Converting regular expression to equivalent NFA.
  - Converting DFA to equivalent regular expression.
Next time

• Are all computational problems decidable?

For Wednesday, pre-class reading: Section 4.3, page 207-209.