Let’s talk decidability.

Code is data!

Working as a TA for 15-112

We need to write an autograder for isPrime

student submission

isPrime

the correct program

isPrime

? Do they return True on exactly the same inputs?

Working as a TA for 15-112

We need to write an autograder for isPrime

Kosbie's version

Student submission

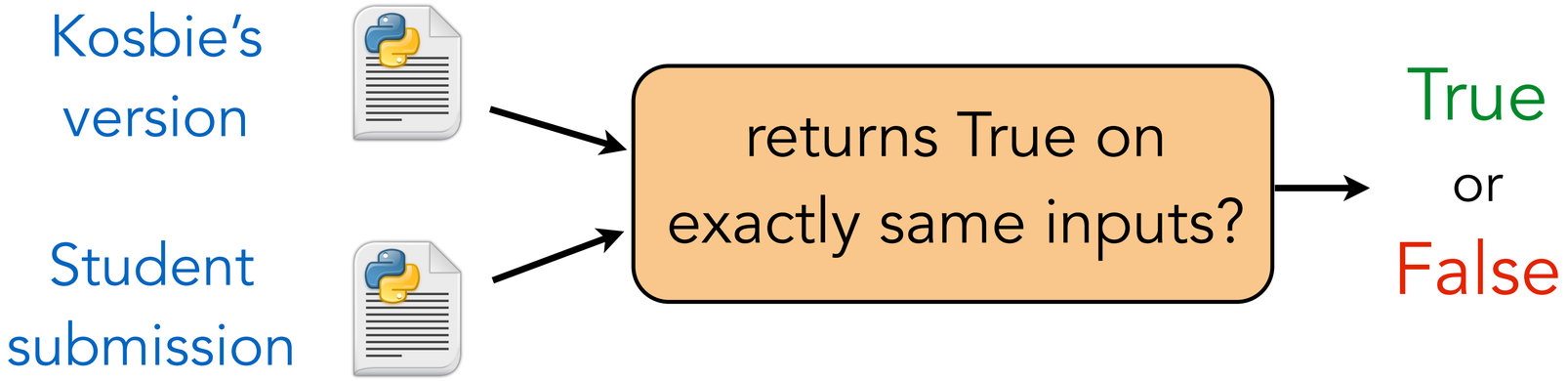
returns True on exactly same inputs?

True

or

False

Working as a TA for 15-112

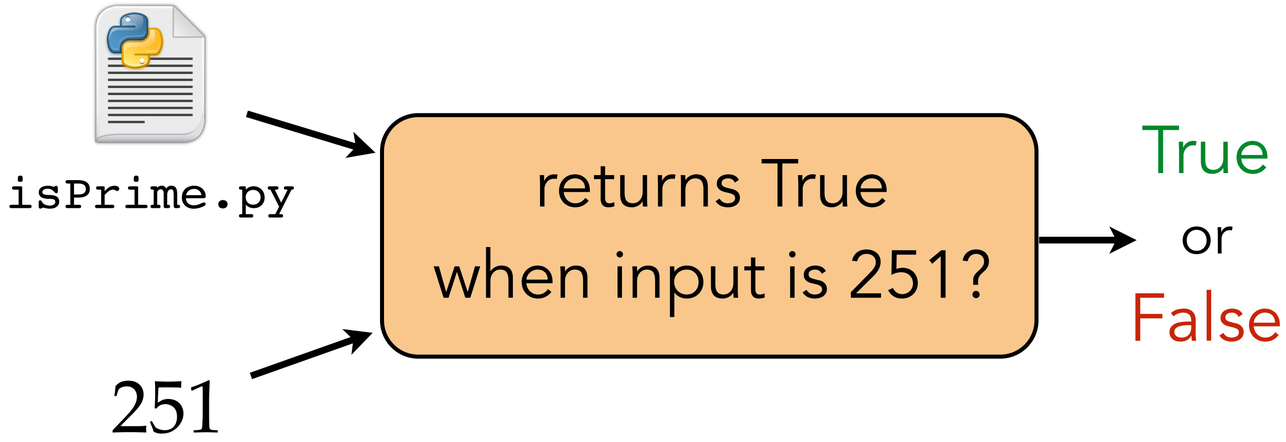


Does such a program exist?

Can we solve/decide the following?

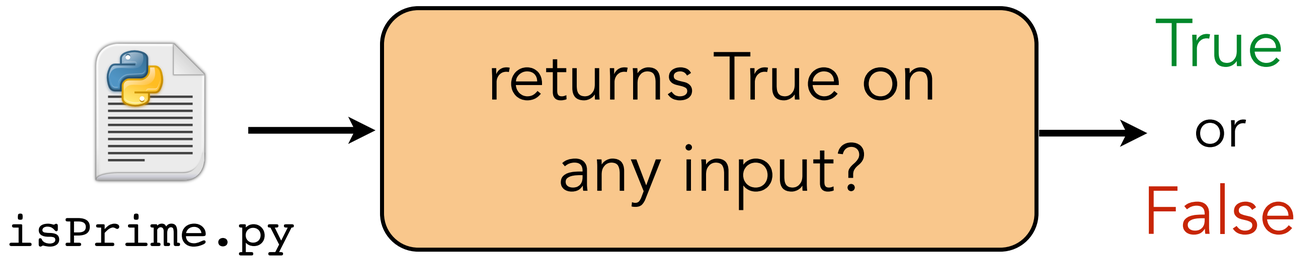
$$\text{NEQ}_{\text{TM}} = \{\langle M_1, M_2 \rangle : M_1, M_2 \text{ are TMs s.t. } L(M_1) \neq L(M_2)\}$$

Working as a TA for 15-112



$$\text{ACCEPTS}_{\text{TM}} = \{\langle M, x\rangle : M \text{ is a TM and } x \in \Sigma^* \text{ s.t. } x \in L(M)\}$$

Working as a TA for 15-112



$$\text{SAT}_{\text{TM}} = \{\langle M \rangle : M \text{ is a TM s.t. } M \text{ accepts some string}\}$$

Poll - which ones are decidable?

$\text{ACCEPTS}_{\text{DFA}} = \{\langle D, x\rangle : D \text{ is a DFA and } x \in \Sigma^* \text{ s.t. } x \in L(D)\}$

$\text{SELF-ACCEPTS}_{\text{DFA}} = \{\langle D\rangle : D \text{ is a DFA and } \langle D\rangle \in L(D)\}$

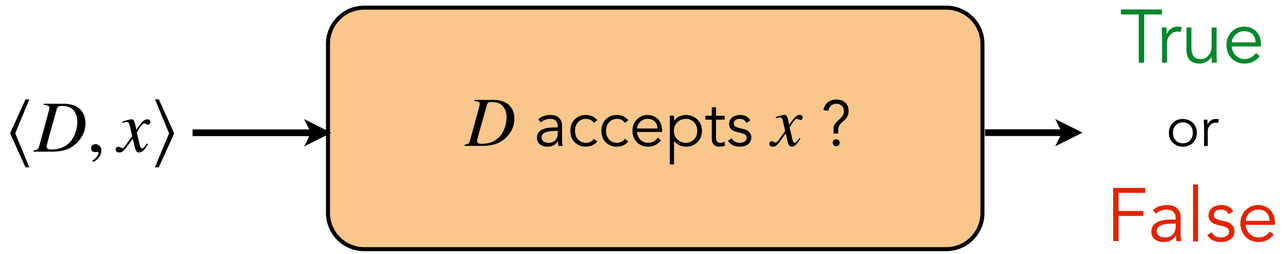
$\text{SAT}_{\text{DFA}} = \{\langle D\rangle : D \text{ is a DFA s.t. } D \text{ accepts some string}\}$

$\text{NEQ}_{\text{DFA}} = \{\langle D_1, D_2\rangle : D_1, D_2 \text{ are DFAs s.t. } L(D_1) \neq L(D_2)\}$

$\text{ACCEPTS}_{\text{DFA}}$

$$\text{ACCEPTS}_{\text{DFA}} = \{\langle D, x\rangle : D \text{ is a DFA and } x \in \Sigma^* \text{ s.t. } x \in L(D)\}$$

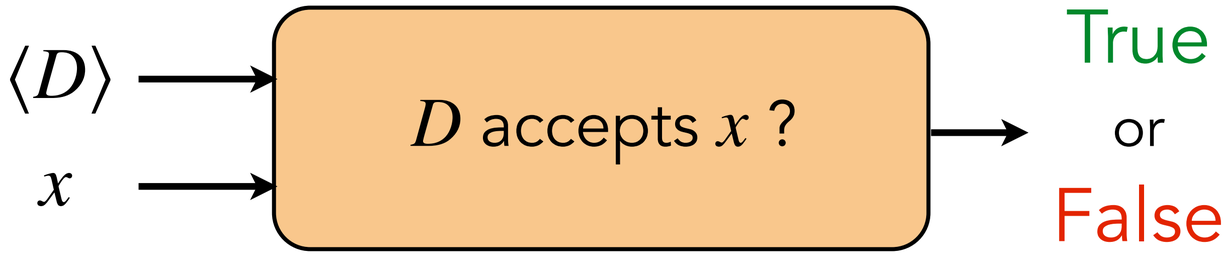
is decidable.



$\text{ACCEPTS}_{\text{DFA}}$

$$\text{ACCEPTS}_{\text{DFA}} = \{\langle D, x\rangle : D \text{ is a DFA and } x \in \Sigma^* \text{ s.t. } x \in L(D)\}$$

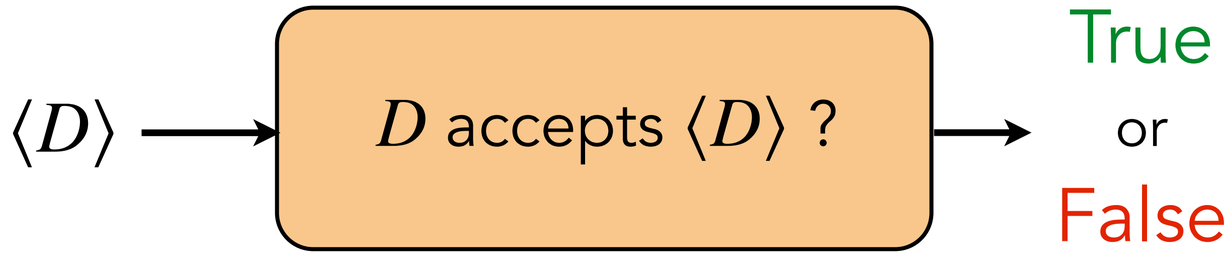
is decidable.



SELF-ACCEPTS$_{DFA}$

$\text{SELF-ACCEPTS}_{\text{DFA}} = \{\langle D \rangle : D \text{ is a DFA and } \langle D \rangle \in L(D)\}$

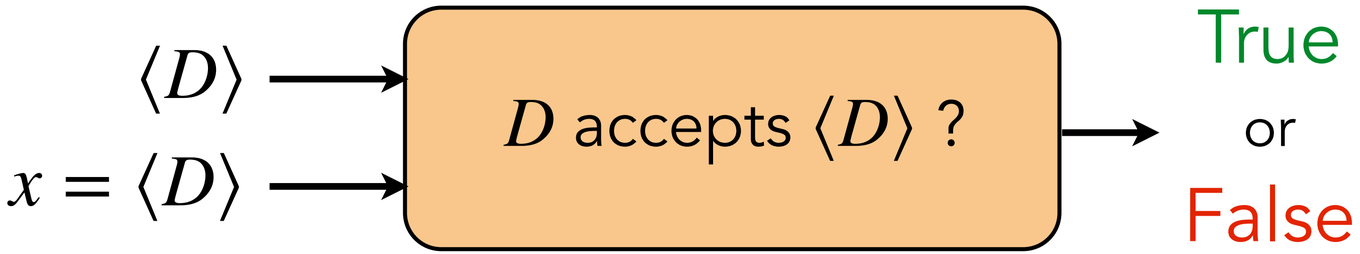
is decidable.



SELF-ACCEPTS$_{DFA}$

$\text{SELF-ACCEPTS}_{\text{DFA}} = \{\langle D \rangle : D \text{ is a DFA and } \langle D \rangle \in L(D)\}$

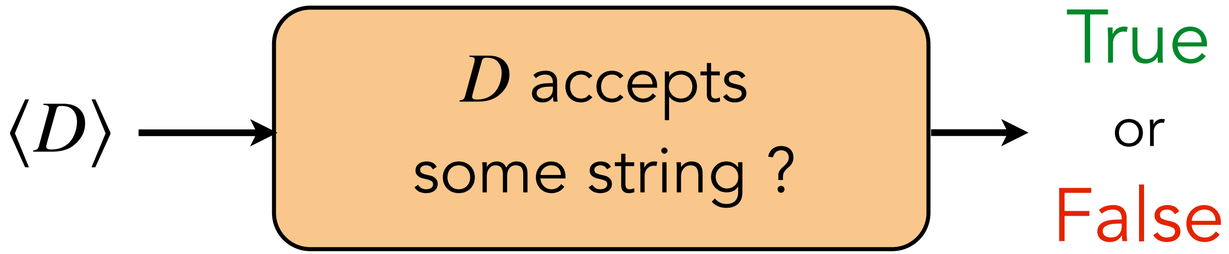
is decidable.



SAT_{DFA}

$\text{SAT}_{\text{DFA}} = \{\langle D \rangle : D \text{ is a DFA s.t. } D \text{ accepts some string}\}$

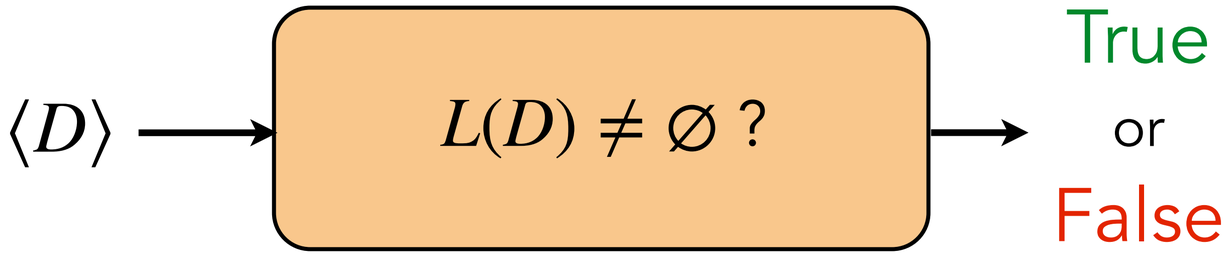
is decidable.



SAT_{DFA}

$\text{SAT}_{\text{DFA}} = \{\langle D \rangle : D \text{ is a DFA s.t. } L(D) \neq \emptyset\}$

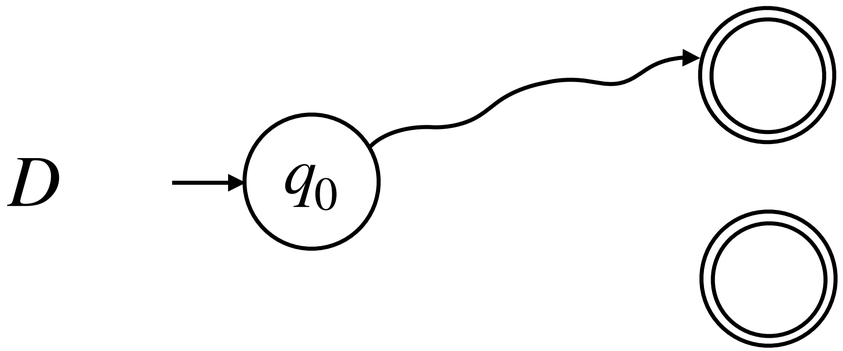
is decidable.



True

or

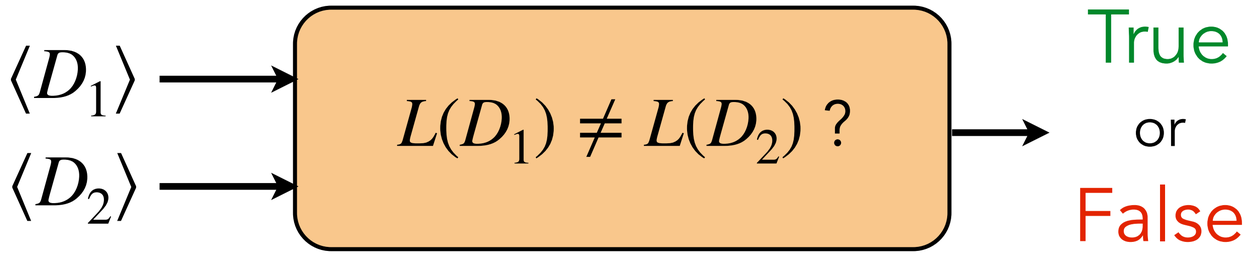
False



NEQ_{DFA}

$\text{NEQ}_{\text{DFA}} = \{\langle D_1, D_2 \rangle : D_1, D_2 \text{ are DFAs s.t. } L(D_1) \neq L(D_2)\}$

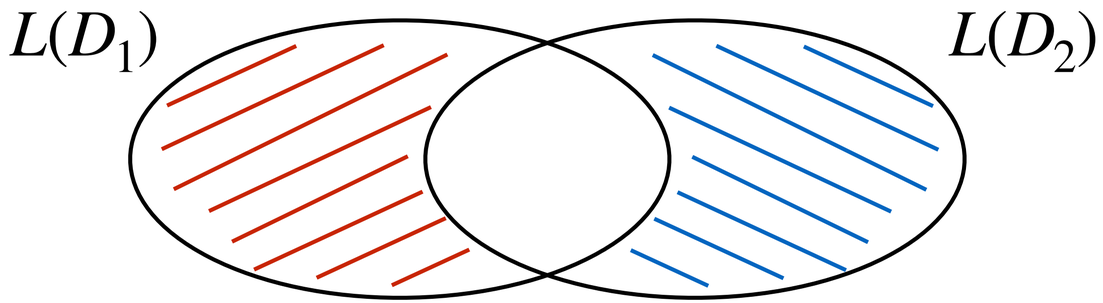
is decidable.



NEQ_{DFA}

$\text{NEQ}_{\text{DFA}} = \{\langle D_1, D_2 \rangle : D_1, D_2 \text{ are DFAs s.t. } L(D_1) \neq L(D_2)\}$

is decidable.



$L(D_1) \neq L(D_2)$ iff shaded region is non-empty.

$\left(L(D_1) \cap \overline{L(D_2)}\right) \cup \left(\overline{L(D_1)} \cap L(D_2)\right) = \varnothing$?

Idea: 1. construct DFA D s.t. $L(D)$ is the shaded region.

2. use algorithm for SAT_{DFA} to determine if $L(D) = \varnothing$.

Reduction

Solving NEQ_{DFA} **reduces** to solving SAT_{DFA}.

NEQ_{DFA} **reduces** to SAT_{DFA}

SAT_{DFA} decidable $\implies$ NEQ_{DFA} decidable.

$\text{NEQ}_{\text{DFA}} \leq \text{SAT}_{\text{DFA}}$

We write $L \leq K$, if we can solve L
using a decider for K as a helper function.

(L is no harder than K)

Reductions expand the landscape of
decidable languages.