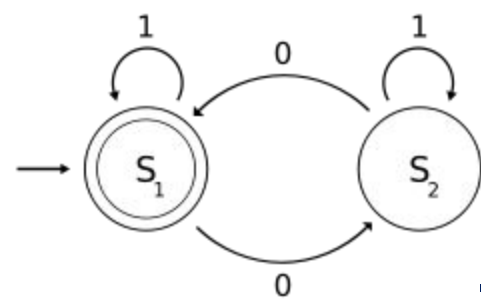
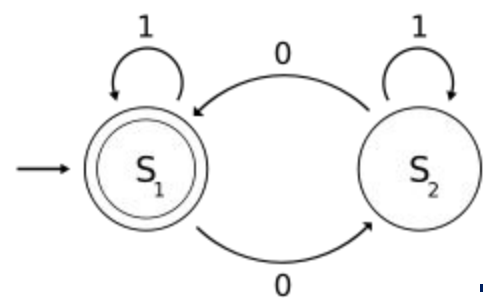




Mapping Reducibility

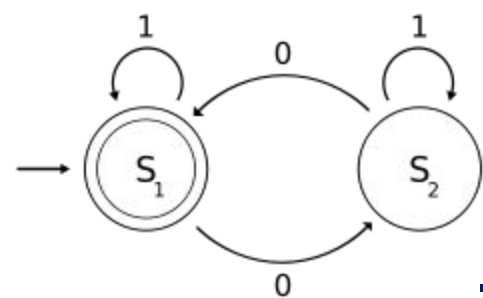


Today's adventures



- formalizing a process we already know
- computable functions
- using problem reduction to show
 - decidability
 - undecidability/
 - recognizability/unrecognizability,
 - co-Turing recognizability & beyond

Weekly tip: There is something we can share with everyone we meet



Warmup

For every TM M , construct the machine:

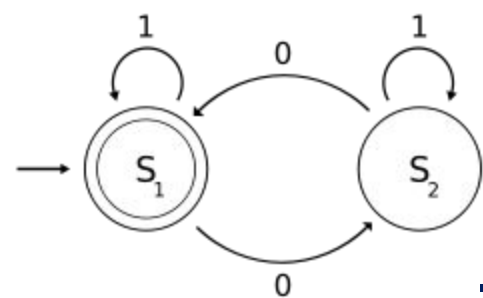
M_∞ = “On input x :

1. Run M on x .
2. If M accepts, *accept*.
3. If M rejects, loop.”

Fill in the blanks:

$L(M_\infty) = \dots$

M_∞ halts if and only if ...



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

Theorem. E_{TM} is undecidable.

Proof. Given an input $\langle M, w \rangle$ we construct a machine M_w that accepts a nonempty language iff M accepts w :

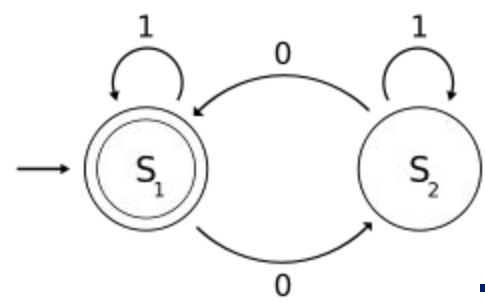
$M_w =$ “On input x :

1. If $x \neq w$, *reject*.
2. If $x = w$, run M on input w and *accept* if M does.”

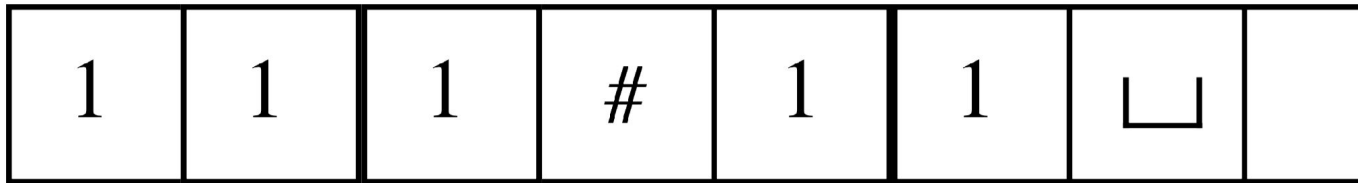
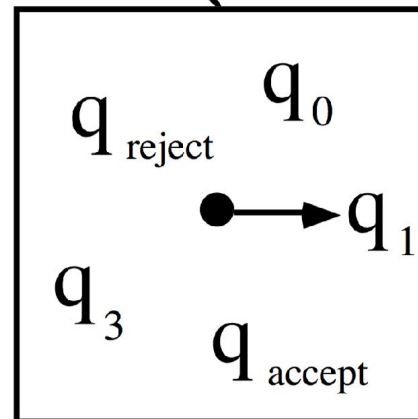
Suppose TM R decides E_{TM} and establish a contradiction by creating a decider S of A_{TM} :

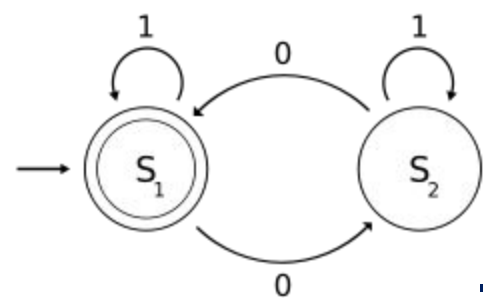
$S =$ “On input $\langle M, w \rangle$:

1. Use the description of M and w to construct M_w .
2. Run R on input $\langle M_w \rangle$.
3. If R accepts, *reject*. If R rejects, *accept*.”



Computable Functions





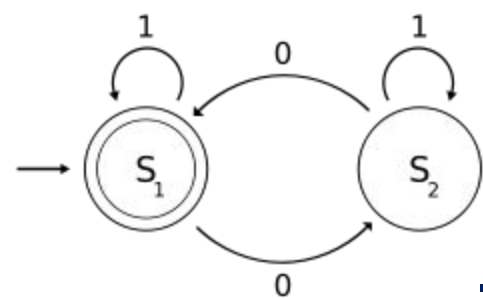
Computable Functions

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

Example. The *increment function*

$$inc^{++}: \{1\}^* \rightarrow \{1\}^*$$

is Turing computable.



Machine Transformers

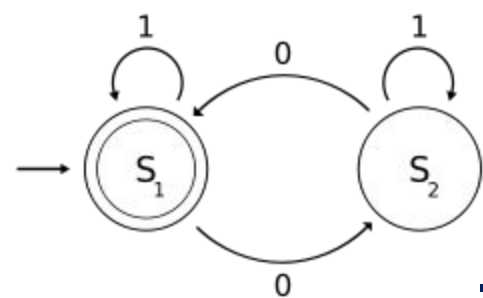
$F =$ “On input $\langle M \rangle$:

1. Construct the machine

$M_\infty =$ “On input x :

1. Run M on x .
2. If M accepts, *accept*.
3. If M rejects, loop.”

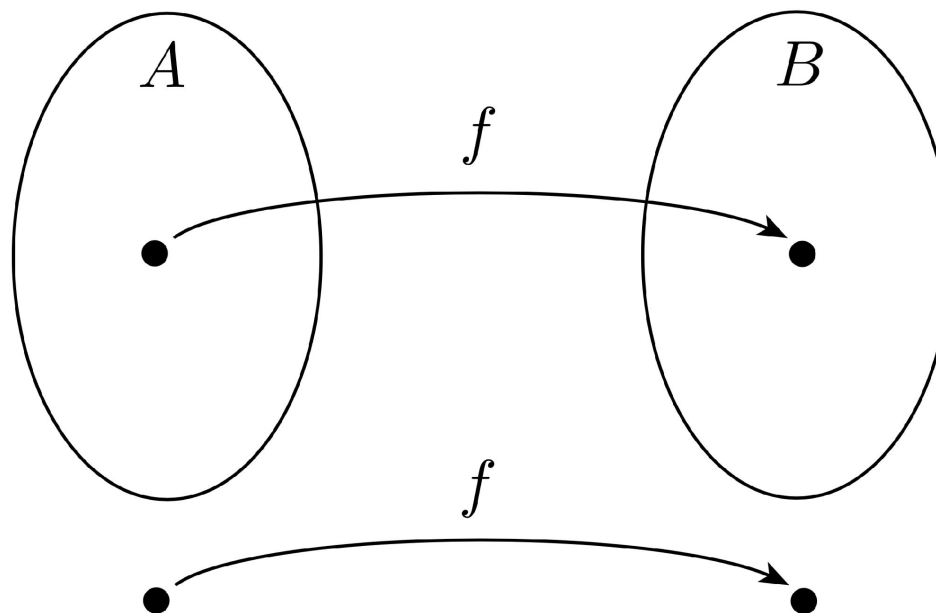
2. Output $\langle M_\infty \rangle$.”

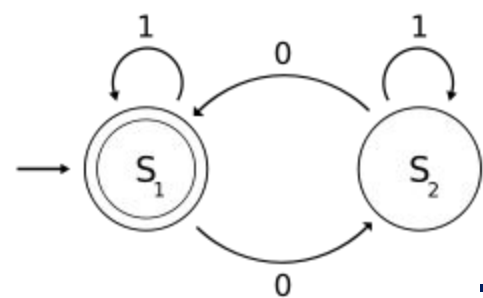


Mapping Reducibility

Definition. Language A is *mapping reducible* to language B , written $A \leq_m B$, if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$, where for every w ,

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

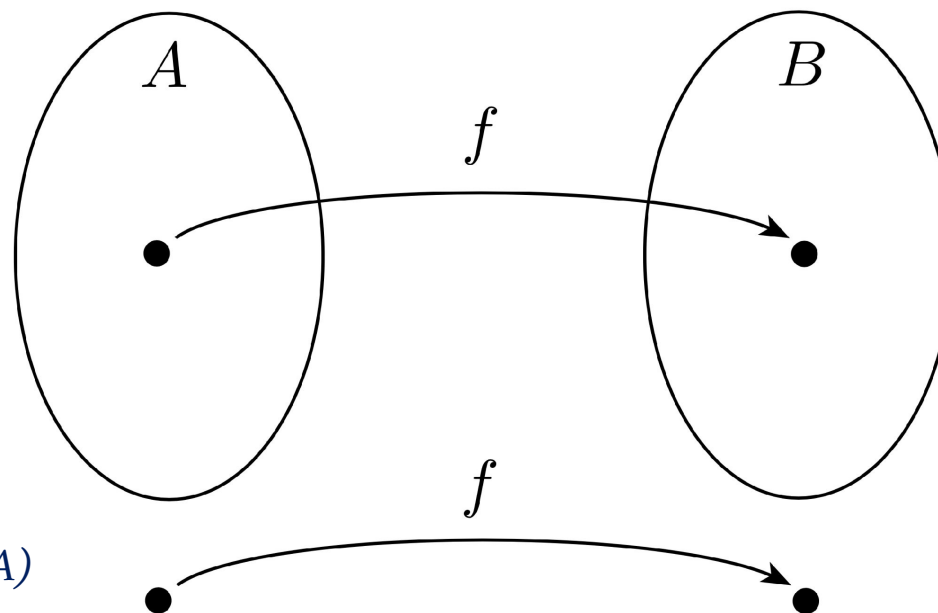




Mapping Reducibility

Definition. Language A is **mapping reducible** to language B , written $A \leq_m B$, if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$, where for every w ,

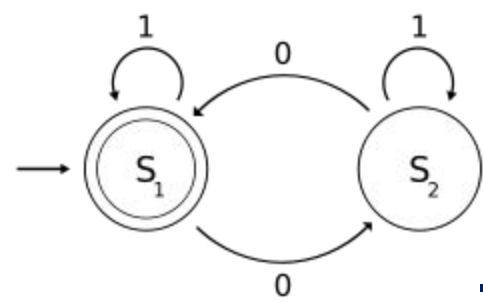
$$w \in A \Leftrightarrow f(w) \in B.$$



$A \leq_m B$
is equivalent to

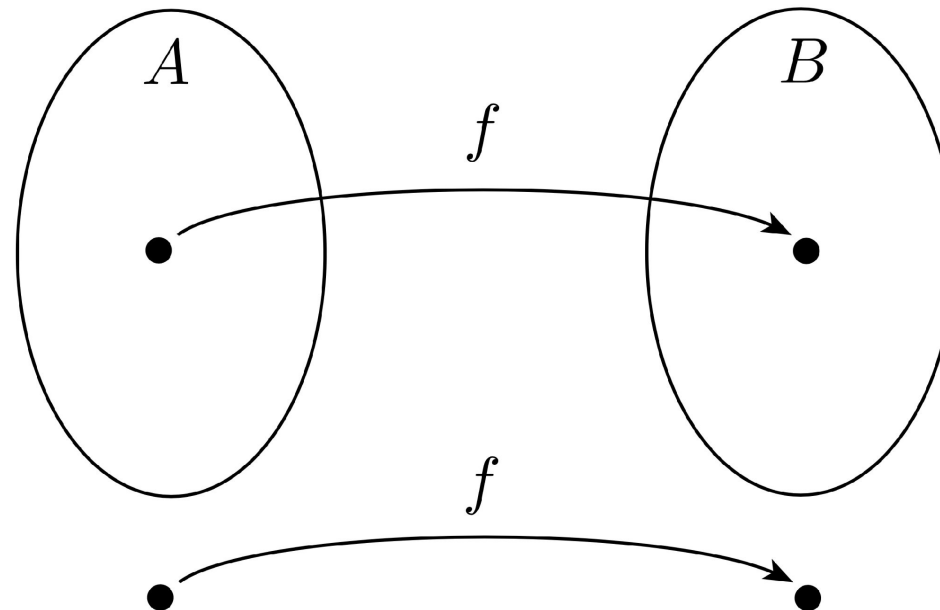
$$A' \leq_m B'$$

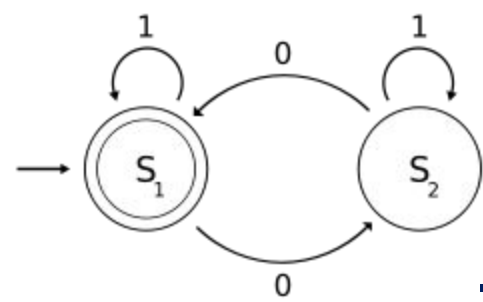
(A' is the complement of A)



Problem Reduction

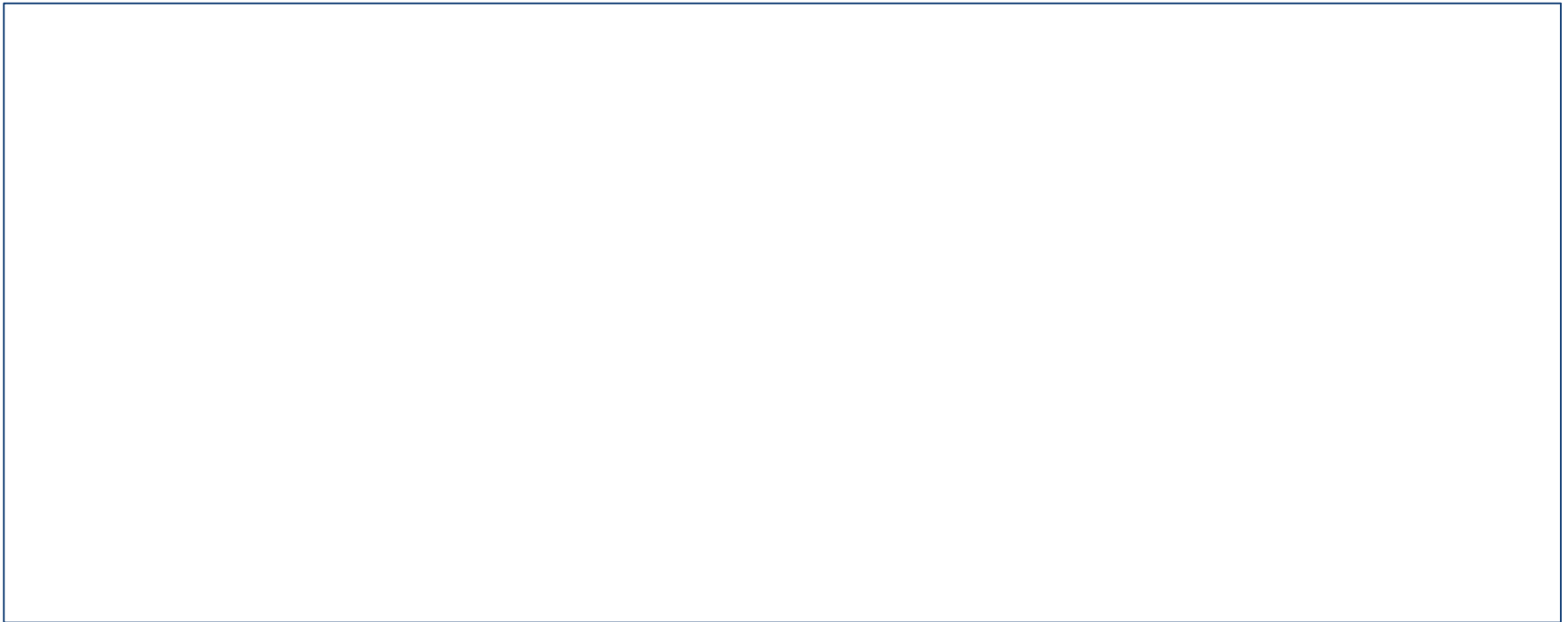
Theorem. If $A \leq_m B$ and B is decidable, then A is decidable.

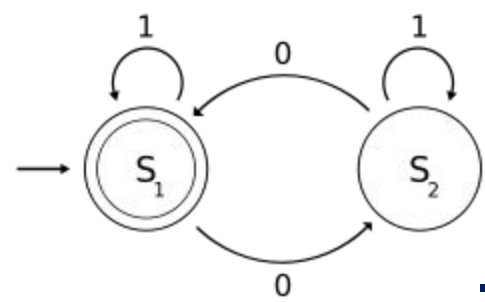




Problem Reduction

Theorem. If $A \leq_m B$ and B is decidable, then A is decidable.

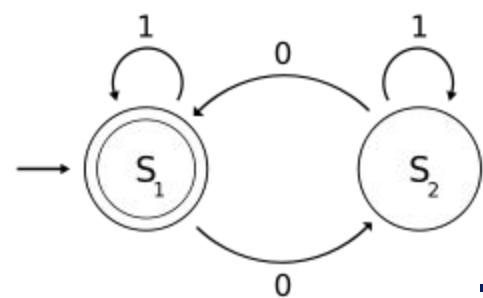




The Contrapositive is Also Useful

Theorem. If $A \leq_m B$ and B is decidable, then A is decidable.

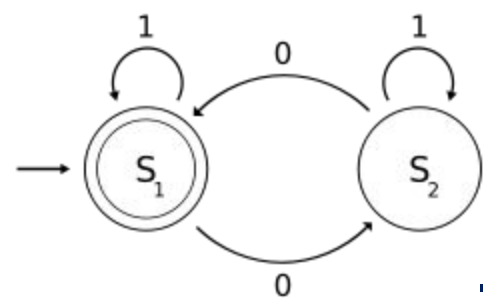
Corollary. If $A \leq_m B$ and A is undecidable, then B is undecidable.



Similarly ...

Theorem. If $A \leq_m B$ and B is Turing-recognizable, then A is Turing-recognizable.

Corollary. If $A \leq_m B$ and A is not Turing-recognizable, then B is not Turing-recognizable.

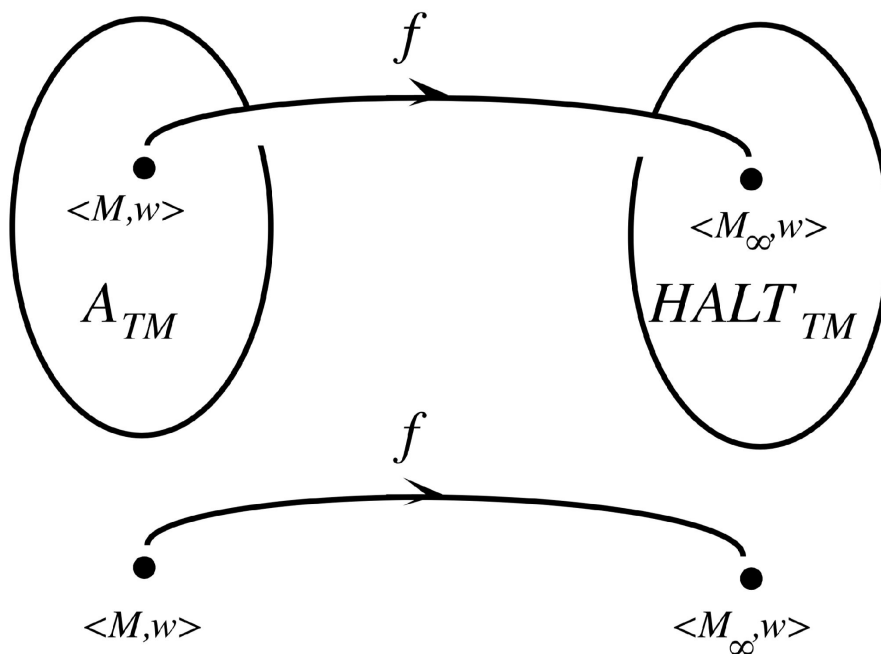


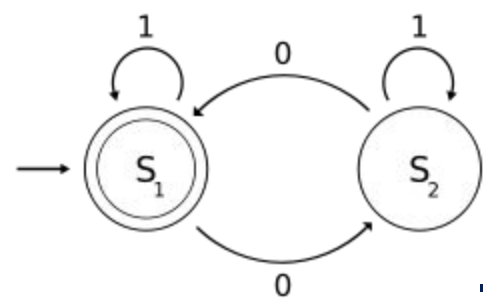
A Familiar Mapping Reduction

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

$$\leq_m$$

$$\text{HALT}_{\text{TM}} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \}$$





$$A_{\text{TM}} \leq_m \text{HALT}_{\text{TM}}$$

$F =$ “On input $\langle M, w \rangle$:

1. Construct the machine

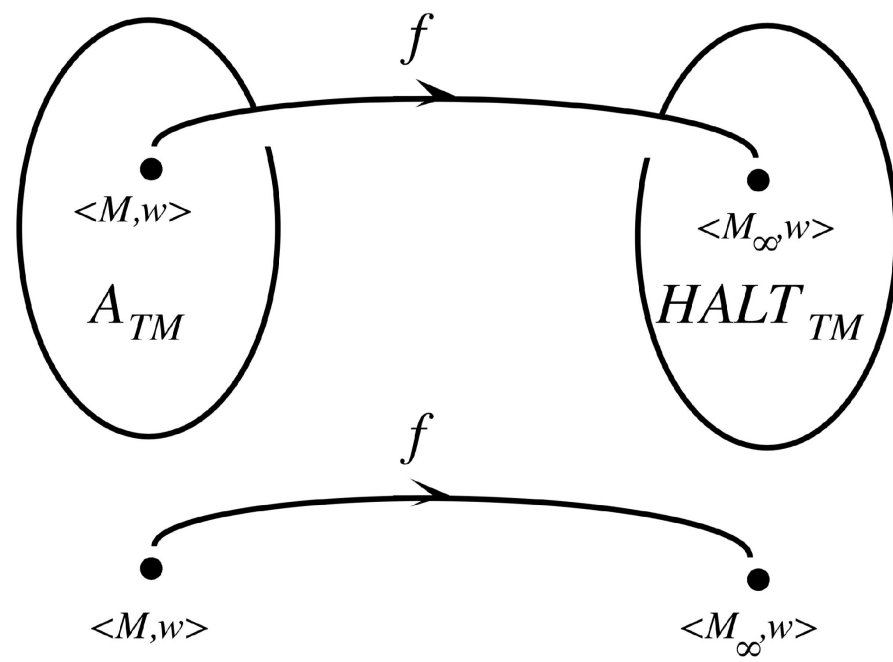
$M_\infty =$ “On input x :

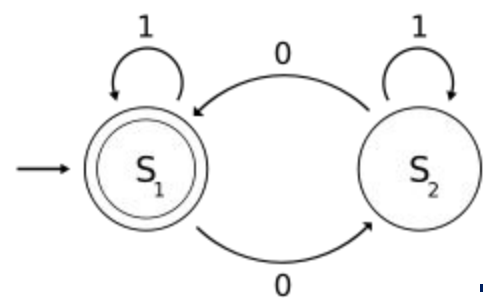
1. Run M on x .

2. If M accepts, *accept*.

3. If M rejects, loop.

2. Output $\langle M_\infty, w \rangle$.”



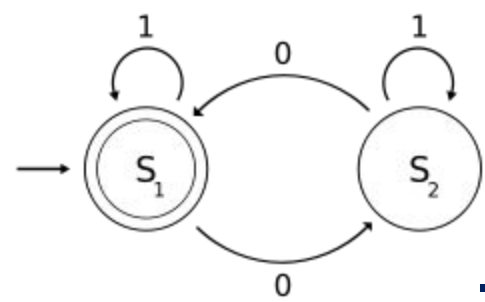


Exercise

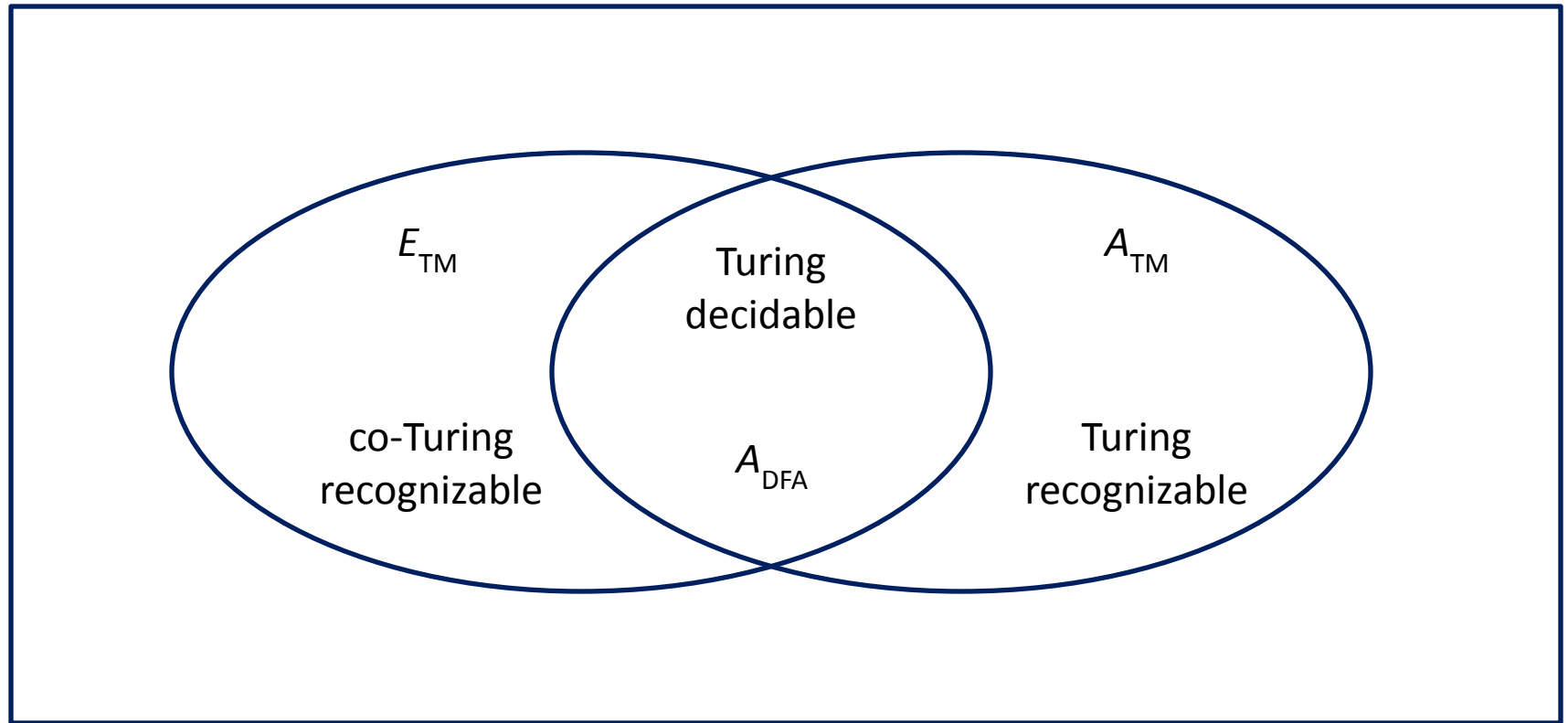
Recall $E_{\text{TM}} = \{ \langle M \rangle \mid M \text{ is a TM \& } L(M) = \emptyset \}$.

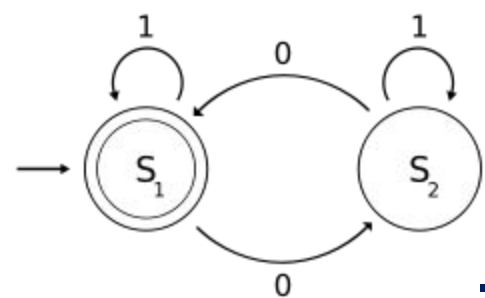
Write out a description of the complement of this language.

Give a description of a Turing-recognizer for the complement of this language.



Solvable, Half-Solvable, Out-to-Lunch

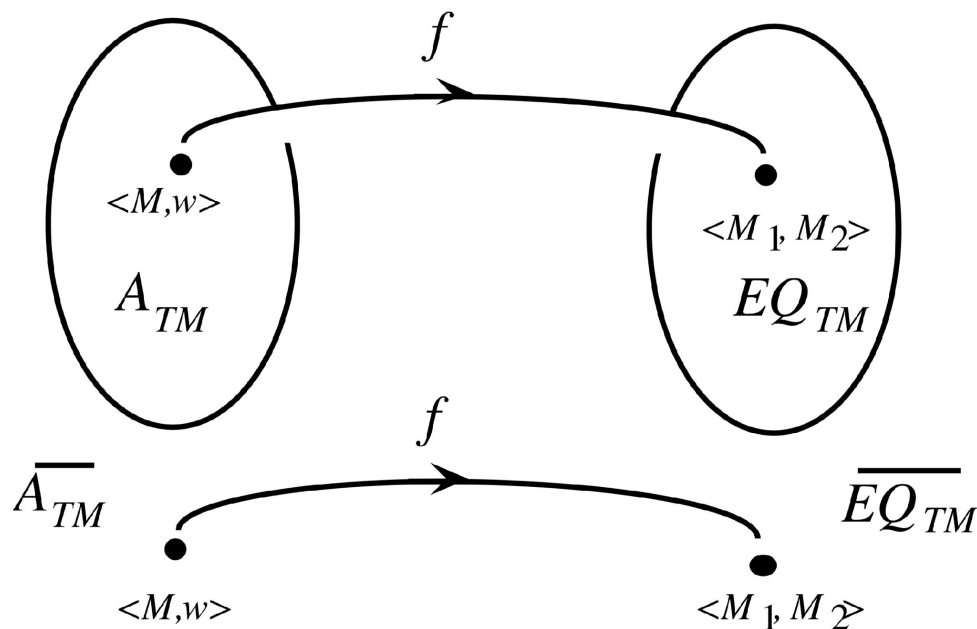


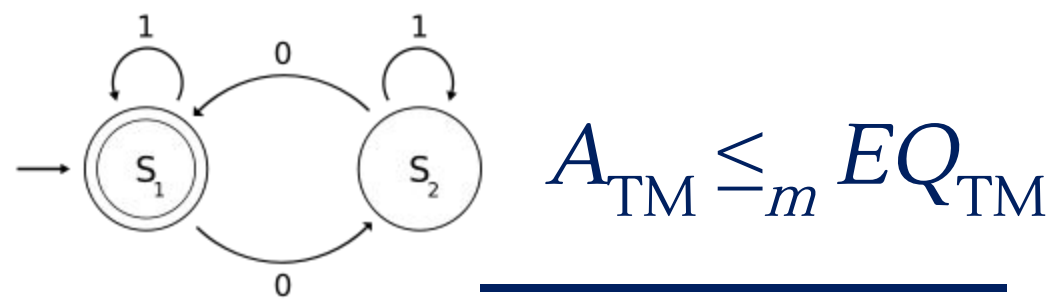
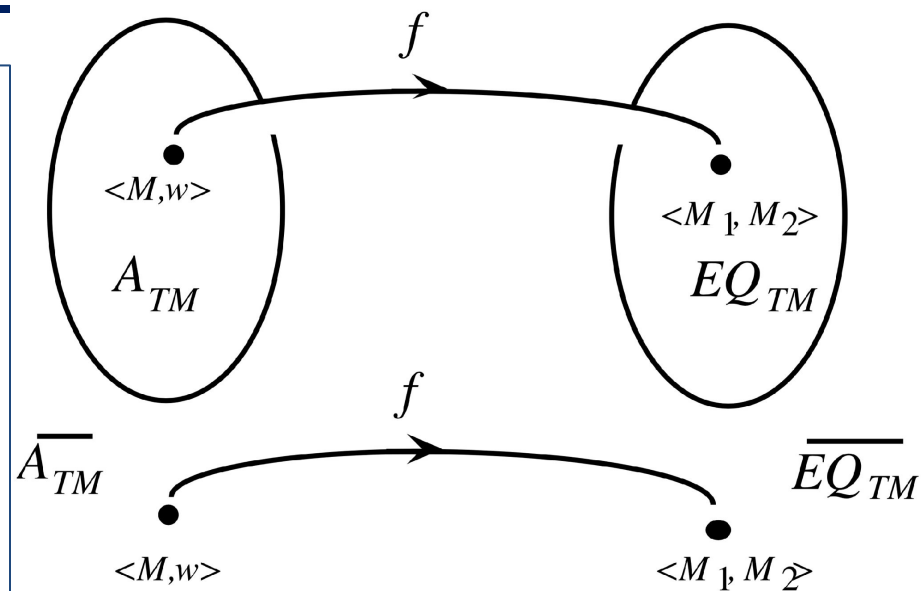


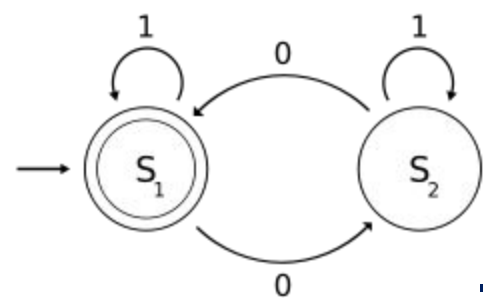
$EQ_{TM} = \{ \langle M_1, M_2 \rangle \mid L(M_1) = L(M_2) \}$
is Out-to-Lunch

Theorem. EQ_{TM} is neither Turing-recognizable nor co-Turing-recognizable.

Proof. We show $A_{TM} \leq_m EQ_{TM}$. (Why does this help?)



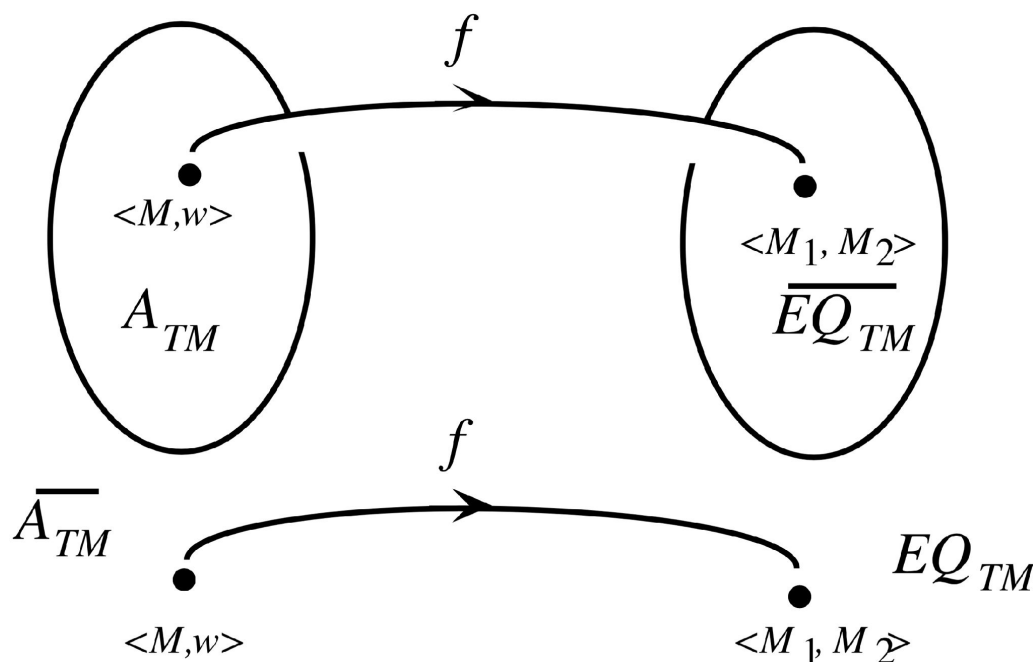


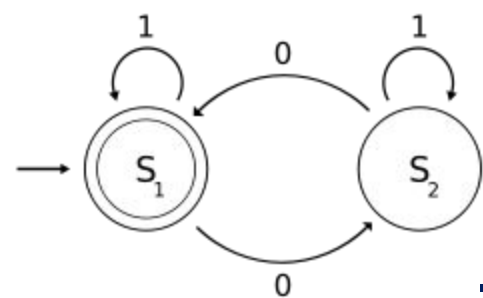


EQ_{TM} is not Turing-recognizable

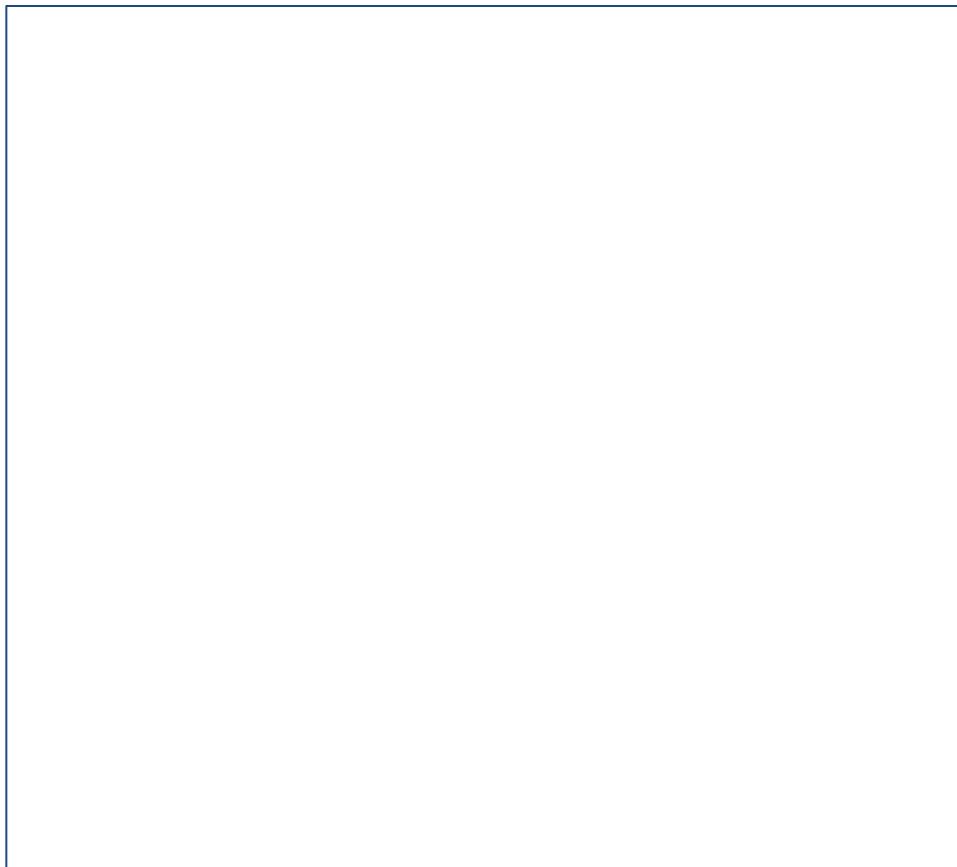
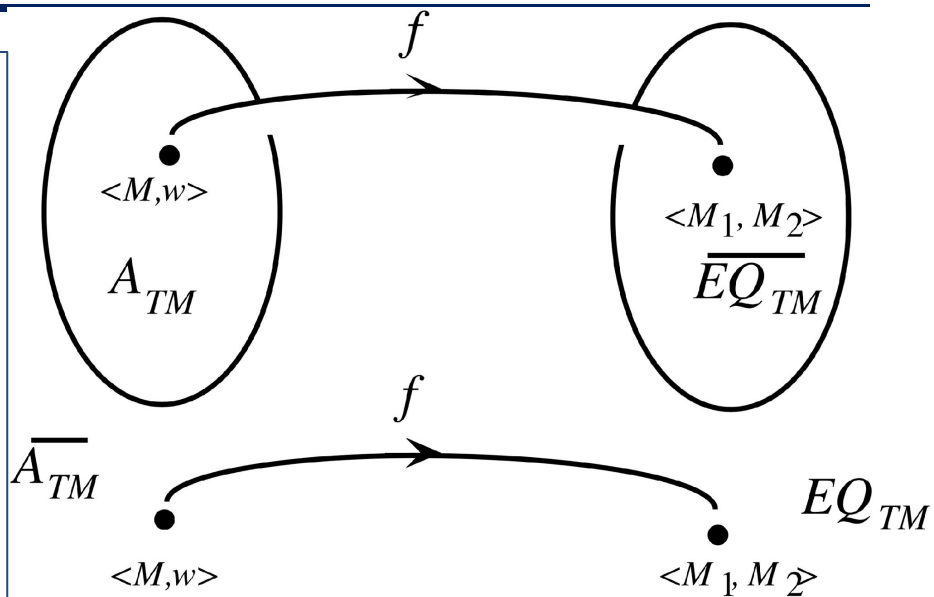
Theorem. EQ_{TM} is neither Turing-recognizable nor co-Turing-recognizable.

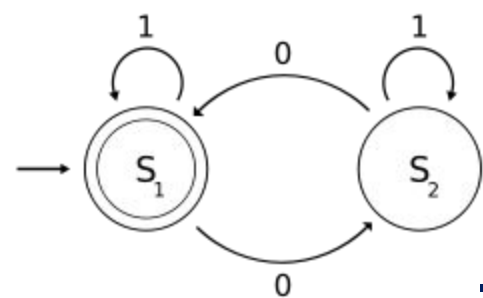
Proof. We show $A_{TM} \leq_m \overline{EQ_{TM}}$. (Why does this help?)





$$A_{TM} \leq_m \overline{EQ_{TM}}$$





Exercises

1. Show that A_{TM} is not mapping reducible to E_{TM} .
 - a. Is $\overline{A_{\text{TM}}}$ Turing recognizable?
 - a. Is $\overline{E_{\text{TM}}}$ Turing recognizable?

2. Show that if P is Turing-recognizable and $P \leq_m \overline{P}$, then P is decidable.