Undecidability II

Lecture 12

Example of Undecidable Language

SELFREJECT = { <M> | *M* rejects <M> }

M =Turing Machine (piece of executable code)

<M> = encoding of M as a string (source code for M)

<M> is what you would feed to a universal TM,

that would allow it to simulate M.

(e.g. TM that rejects everything.

TM that rejects every description of a TM are in that language

Showing Undecidability

o show L is undecidable, reduce some undecidable language to

$SELFHALT = \{$ <M> | M halts on <M> $\}$

Claim: *SELFHALT* is undecidable

More general looking problem:

$HALT = \{$ <M,w> | M halts on w $\}$

Claim: *HALT* is acceptable

The halting problem

Claim: *HALT* is undecidable

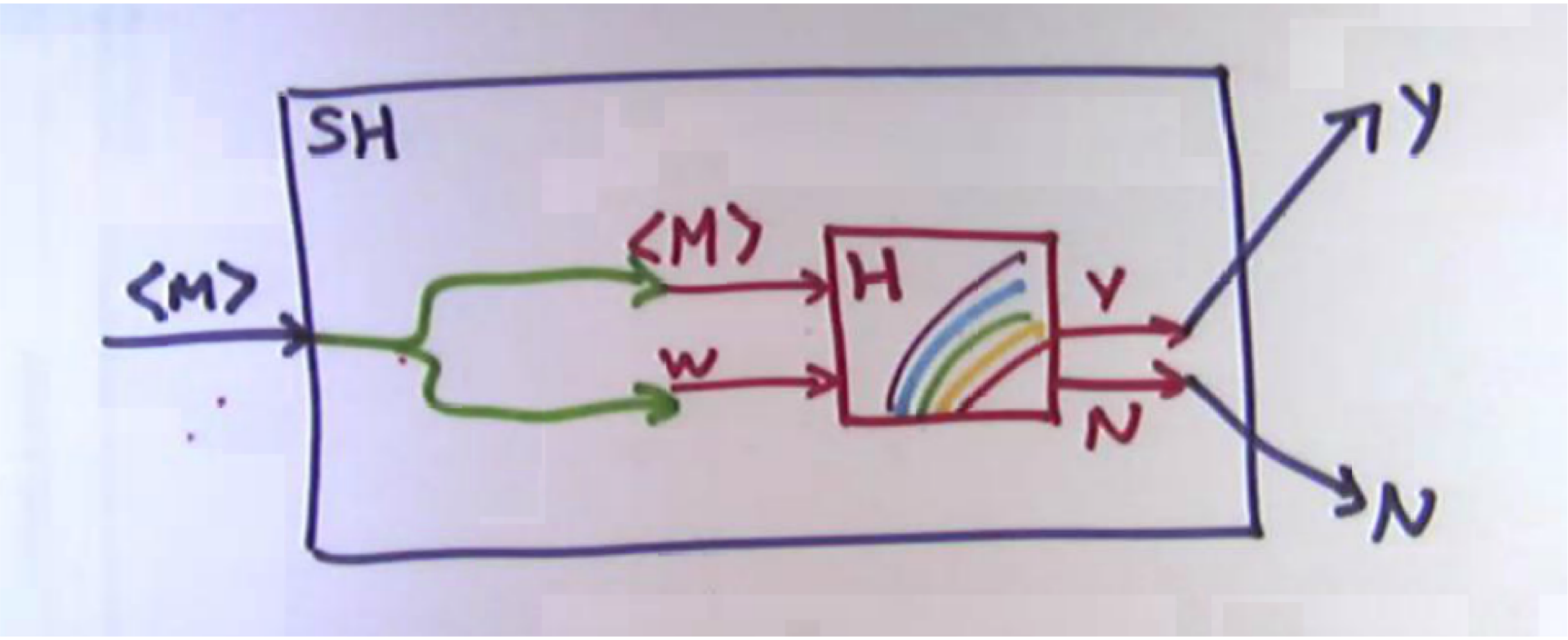
Showing Undecidability

HALT = { <M,w> | *M* halts on w }

Claim: *HALT* is undecidable

Proof:

Suppose (towards contradiction) that there is a TM H that decides HALT. Reduce from SELFHALT



NEVER*ACCEPT* = { <M> | *ACCEPT(M)*=ϕ }

(is a TM useless or not?)

Claim: NEVERACCEPT is undecidable

How many Turing Machines?

- Fix a TM M and an input w.
- Build a new TM M’ with the following behavior:
- M’ accepts its input iff M accepts w. (toss input out the window)
- Pseudocode :

M’(x)

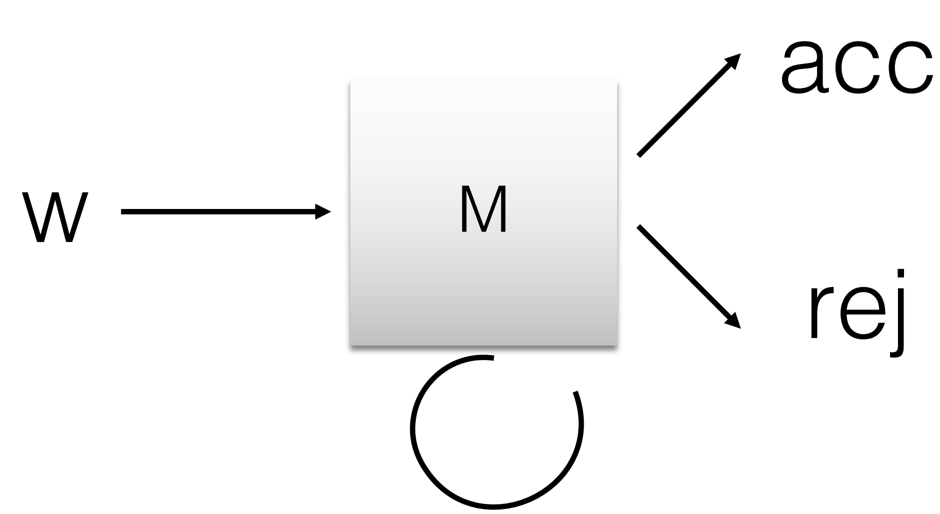
Run M(w)

How many Turing Machines?

- Fix a TM M and an input w.
- Build a new TM M’ with the following behavior:
- M’ accepts its input iff M accepts w. (toss input out the window)
- Pseudocode :

M’(x)

Run M(w)

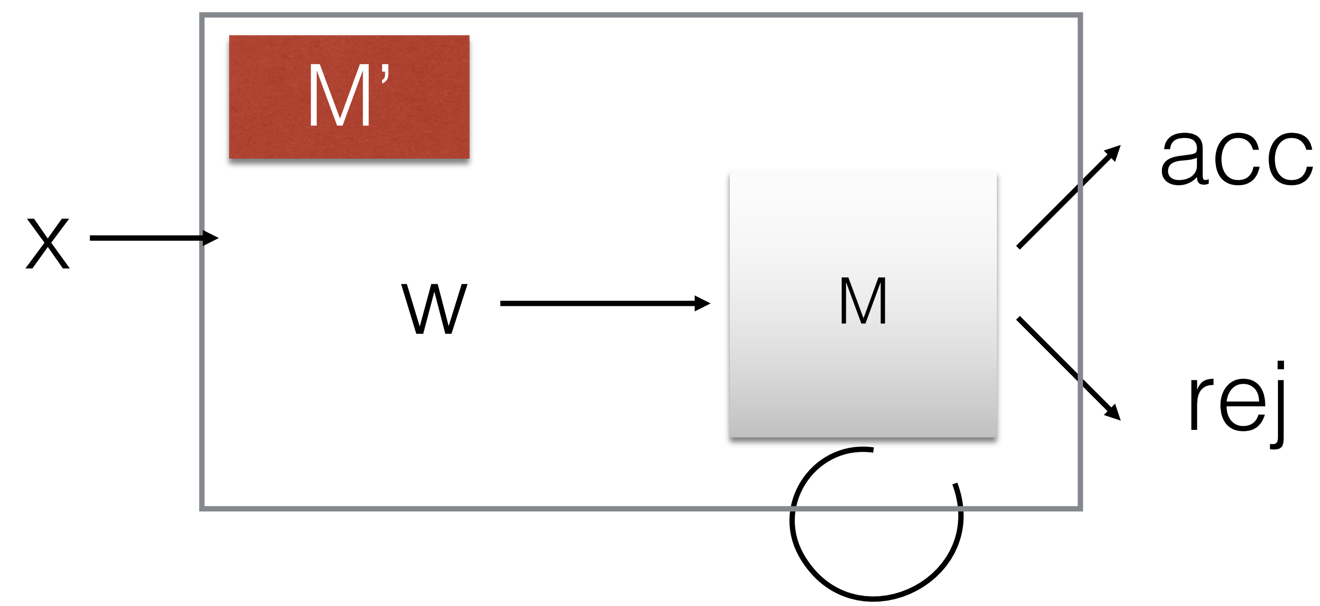


How many Turing Machines?

- Fix a TM M and an input w.
- Build a new TM M' with the following behavior:
- M' accepts its input iff M accepts w. (toss input out the window)
- Pseudocode :

M'(x)

Run M(w)



w hardcoded and M hardcoded in M'

- Build M'?

Write a program

Input <M,w>: M - Turing Machine,

w - string

Output <M'>: M' - Turing Machine,

s.t. for any string x, M' accepts x iff M accepts w.

- could produce M' ourselves (write pseudocode).

So far, when we talk about reduction, WE are doing the reductio

- Now, we need to describe how to do this transformation
  - by writing code that performs the transformation

NEVER*ACCEPT* = { <M> | *ACCEPT(M)*=ϕ }

(M accepts nothing)

Claim: NEVERACCEPT is undecidable

We will assume we know the following:

ACCEPT = { <M,w> | *M accepts w* } is undecidable

Proof:

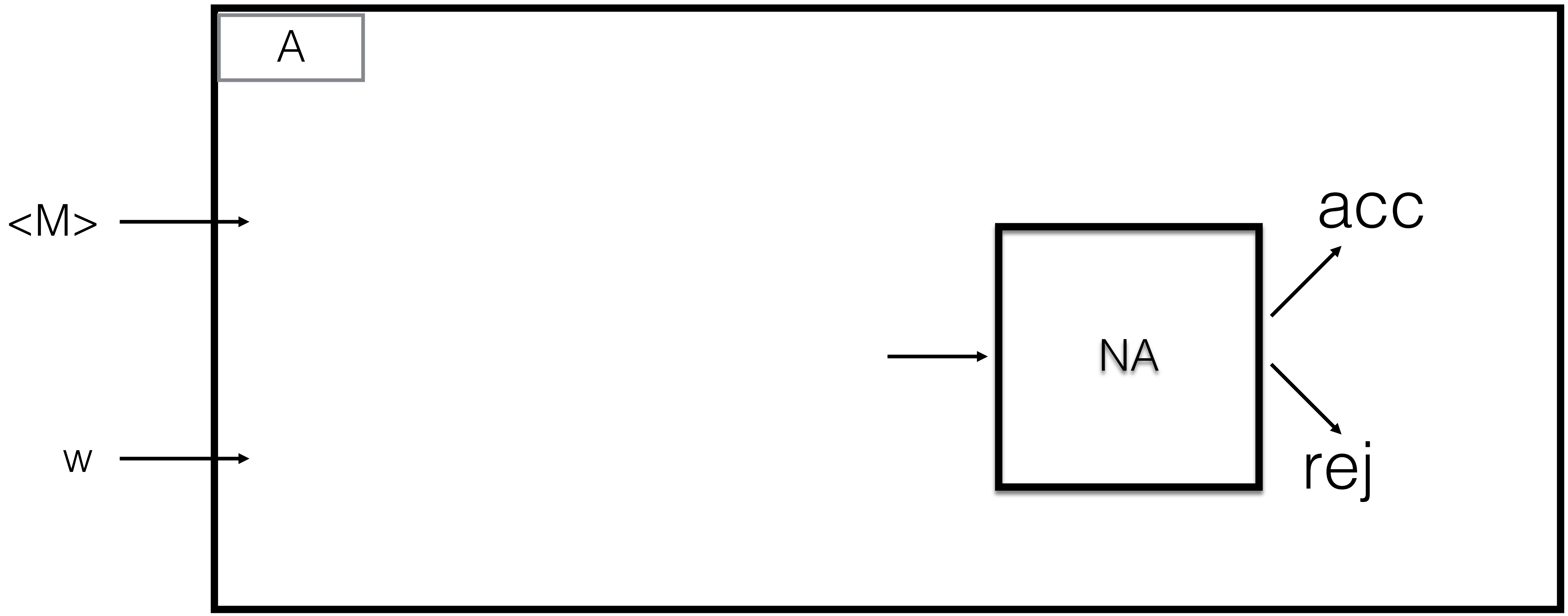
Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.

$$NEVERACCEPT = \{ \langle M \rangle \mid ACCEPT(M) = \emptyset \}$$

Claim: NEVERACCEPT is undecidable

Proof:

Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.

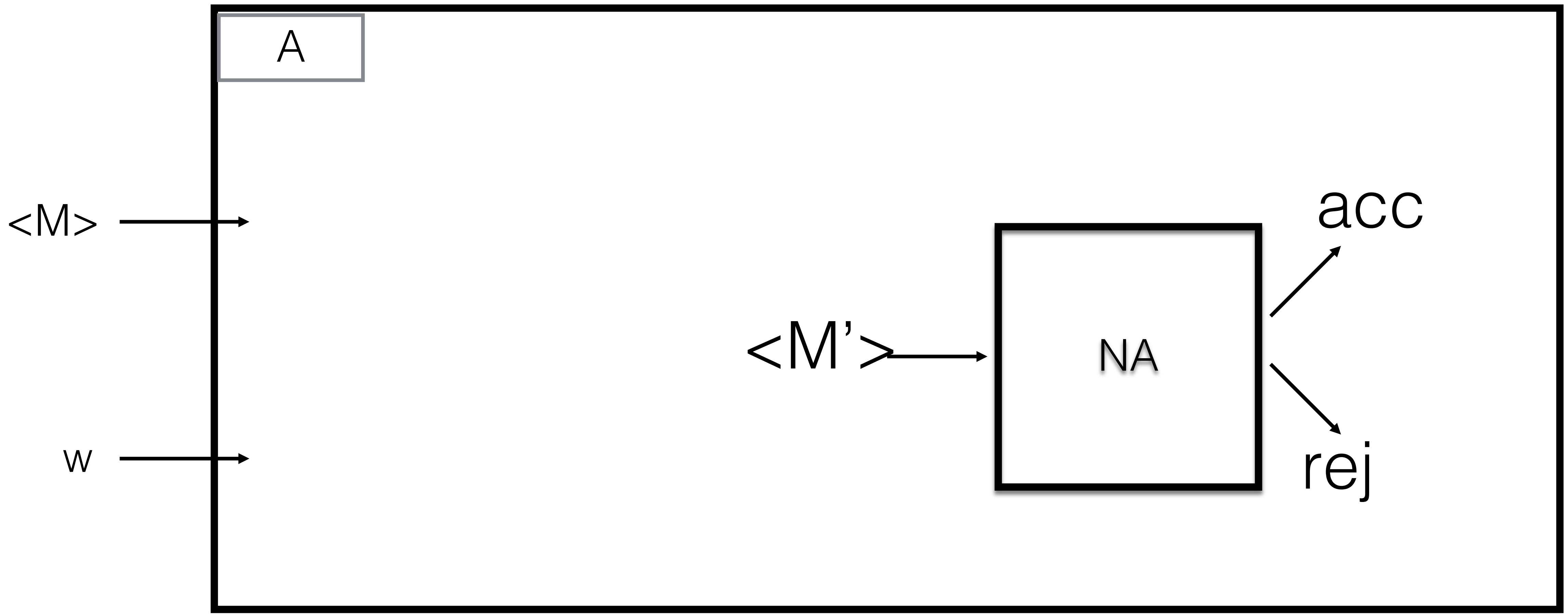


$$NEVERACCEPT = \{ \langle M \rangle \mid ACCEPT(M) = \emptyset \}$$

Claim: NEVERACCEPT is undecidable

Proof:

Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.



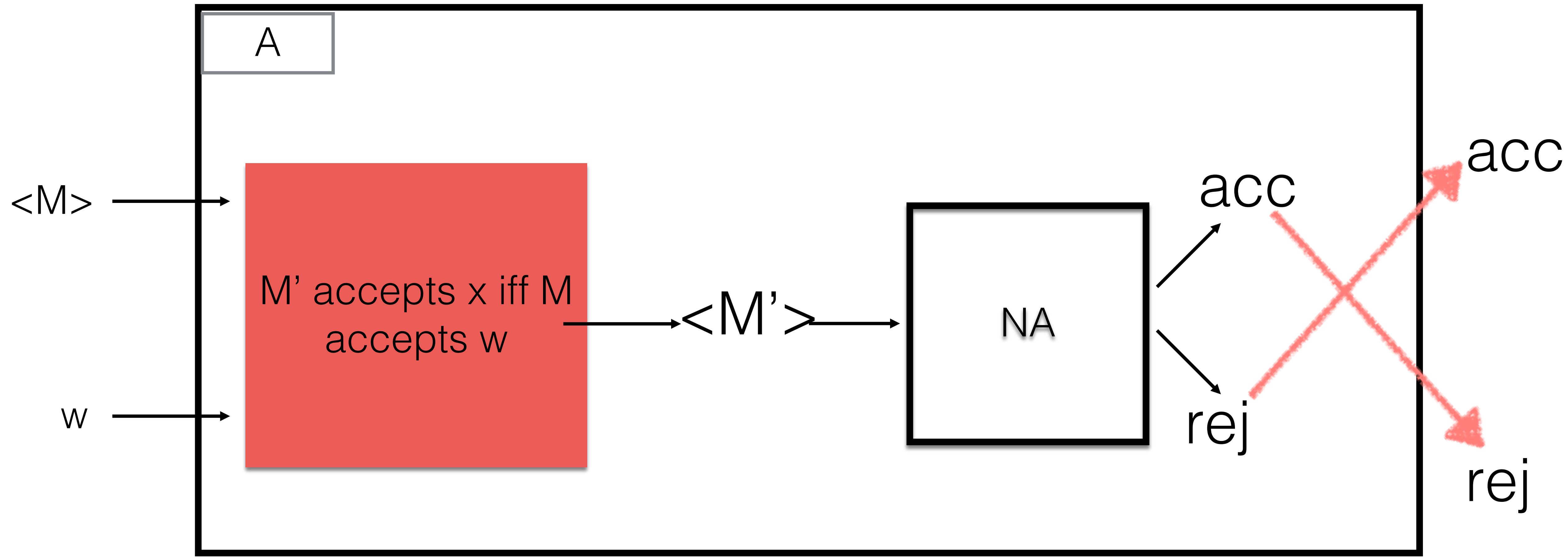
NEVER$ACCEPT$ = { <M> | $ACCEPT(M)=\phi$ }

Claim: NEVERACCEPT is undecidable

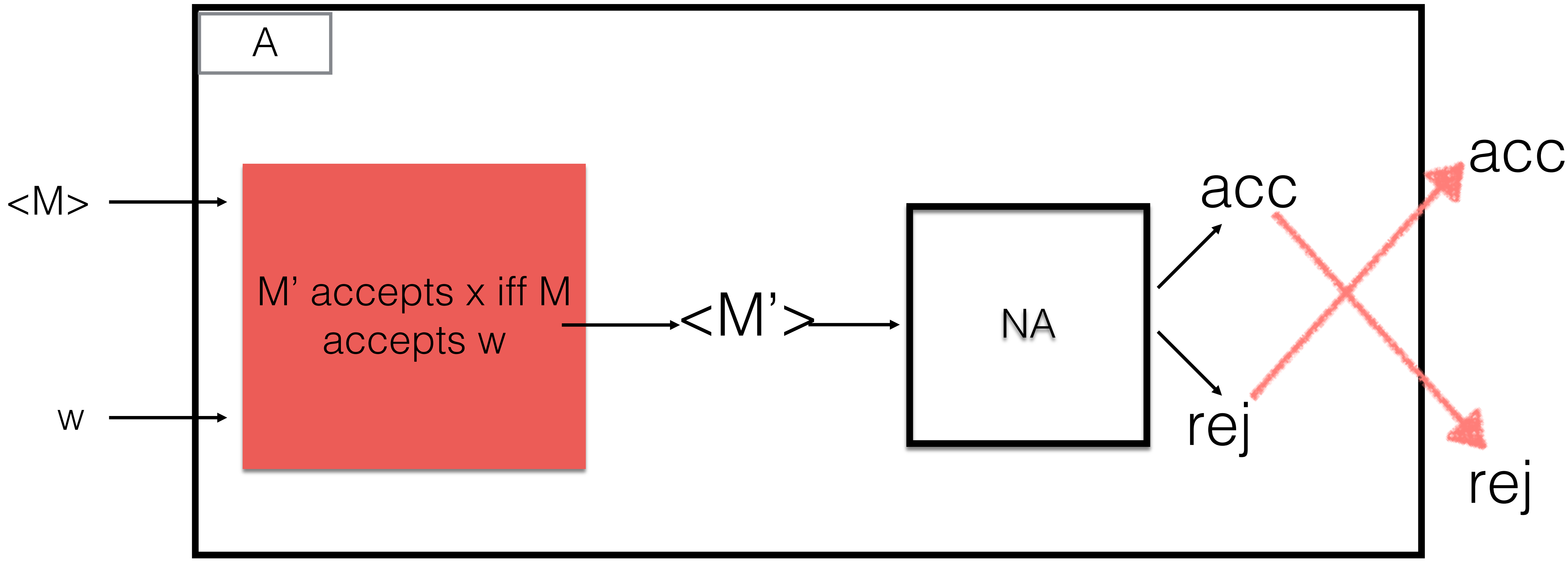
Proof:

Suppose (towards contradiction) that there is a TM NA that decides NEVERACCEPT.

how many TMs?



when I design a compiler for a piece of code,
I can't worry about the input that this code will be fed many many years from now.
x and w not related!



NEVER$ACCEPT$ = { <M> | $ACCEPT(M)=\phi$ }

Claim: A decides ACCEPT

- **Case 1**: M accepts w.

NEVER$ACCEPT$ = { <M> | $ACCEPT(M)=\emptyset$ }

Claim: A decides ACCEPT

- **Case 1**: M accepts w.

Implies M’ accepts everything (by def. of M’).

Implies M’ not in NEVERACCEPT (by def of NEVERACCEPT)

Implies NA rejects <M’> (by def of NA)

Implies A accepts <M,w> (by def of A)

NEVER$ACCEPT$ = { <M> | $ACCEPT(M)=\phi$ }

Claim: A decides ACCEPT

- **Case 2**: M doesn’t accept w.

NEVER*ACCEPT* = { <M> | *ACCEPT(M)*=ϕ }

Claim: A decides ACCEPT

- **Case 2**: M doesn't accept w.

Implies M' doesn't accept anything (by def. of M').

Implies M' in NEVERACCEPT (by def of NEVERACCEPT)

Implies NA accepts <M'> (by def of NA)

Implies A rejects <M,w> (by def of A)

These two cases are exhaustive and imply A decides ACCEPT, contradiction

$$\textsc{NeverAccept} := \{\langle M \rangle \mid \textsc{Accept}(M) = \varnothing\}$$

$$\textsc{NeverReject} := \{\langle M \rangle \mid \textsc{Reject}(M) = \varnothing\}$$

$$\textsc{NeverHalt} := \{\langle M \rangle \mid \textsc{Halt}(M) = \varnothing\}$$

$$\textsc{NeverDiverge} := \{\langle M \rangle \mid \textsc{Diverge}(M) = \varnothing\}$$

DIVERGERSAME = { <M1><M2> | *DIVERGE(M1) =DIVERGE(M2)* }

Claim: Undecidable

Theorem 14. *The language* $\textsc{DivergeSame} := \left\{ \langle M_1 \rangle \langle M_2 \rangle \;\middle|\; \textsc{Diverge}(M_1) = \textsc{Diverge}(M_2) \right\}$ *is undecidable.*

Proof: Suppose for the sake of argument that there is a Turing machine DS that decides DIVERGESAME. Then we can build a Turing machine ND that decides NEVERDIVERGE as follows. Fix a Turing machine Y that accepts Σ^* (for example, by defining $\delta(\text{start}, a) = (\text{accept}, \cdot, \cdot)$ for all $a \in \Gamma$). Given an arbitrary Turing machine encoding $\langle M \rangle$ as input, ND writes the string $\langle M \rangle \langle Y \rangle$ onto the tape and then passes control to DS. There are two cases to consider:

- If DS accepts $\langle M \rangle \langle Y \rangle$, then $\textsc{Diverge}(M) = \textsc{Diverge}(Y) = \varnothing$, so $\langle M \rangle \in \textsc{NeverDiverge}$.
- If DS rejects $\langle M \rangle \langle Y \rangle$, then $\textsc{Diverge}(M) \neq \textsc{Diverge}(Y) = \varnothing$, so $\langle M \rangle \notin \textsc{NeverDiverge}$.

In short, ND accepts $\langle M \rangle$ if and only if $\langle M \rangle \in \textsc{NeverDiverge}$, which is impossible. We conclude that DS does not exist. □

Rice's Theorem

- We want to answer questions of the form "does the language this machine accepts have some interesting property?"
- L={set of acceptable languages that is not empty and is not the set of all languages}
    - e.g. L = set of all languages containing the word "surfing"
        - Define ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}
- L =ø : ACCEPTIN(ø) is decidable (always say no, no language is element of ø)
- L= everything: ACCEPTIN(all) is decidable (always say yes: does this TM accept a language?)
    - For every other L ACCEPTIN(L) is undecidable

Rice's Theorem

Rice's Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that $\textsc{Accept}(Y) \in \mathcal{L}$.*
- *There is a Turing machine N such that $\textsc{Accept}(N) \notin \mathcal{L}$.*

The language $\textsc{AcceptIn}(\mathcal{L}) := \left\{ \langle M \rangle \;\middle|\; \textsc{Accept}(M) \in \mathcal{L} \right\}$ *is undecidable.*

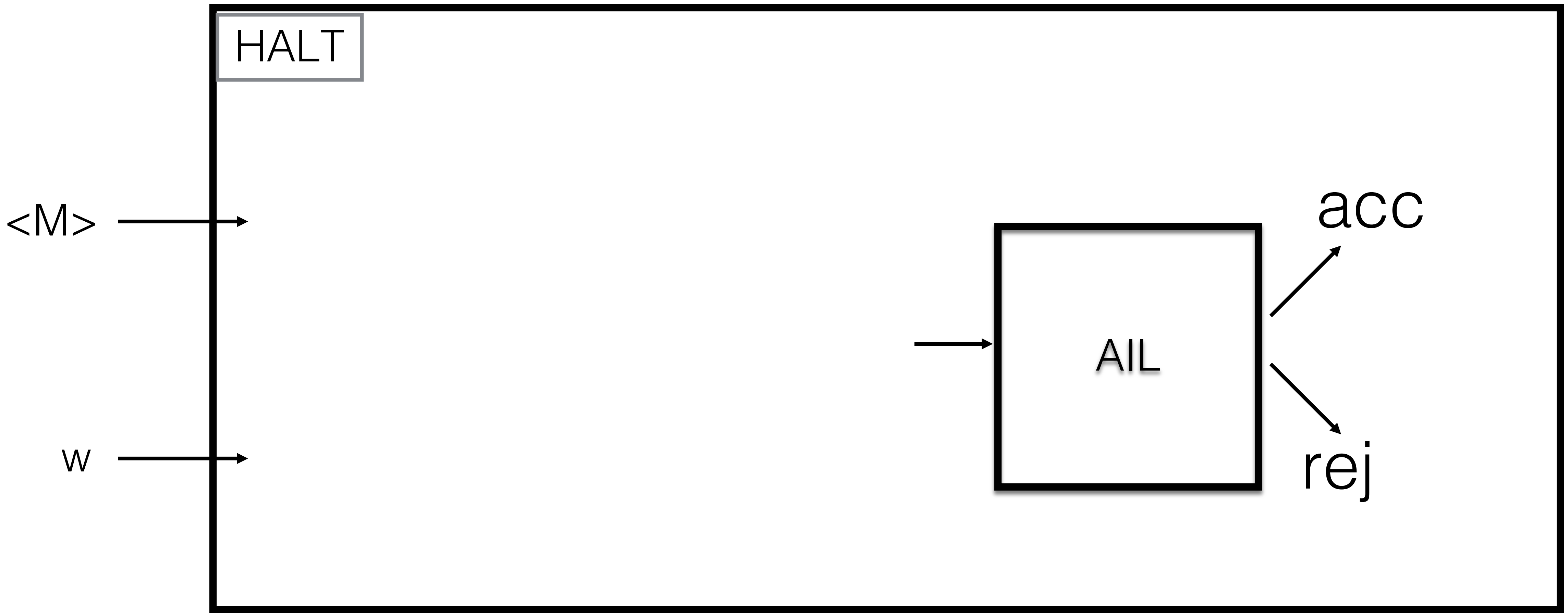
To Show ACCEPTIN(L) is undecidable

Reduce from *HALT* = { <M,w> | *M* halts on w }

Rice’s Theorem

- ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}

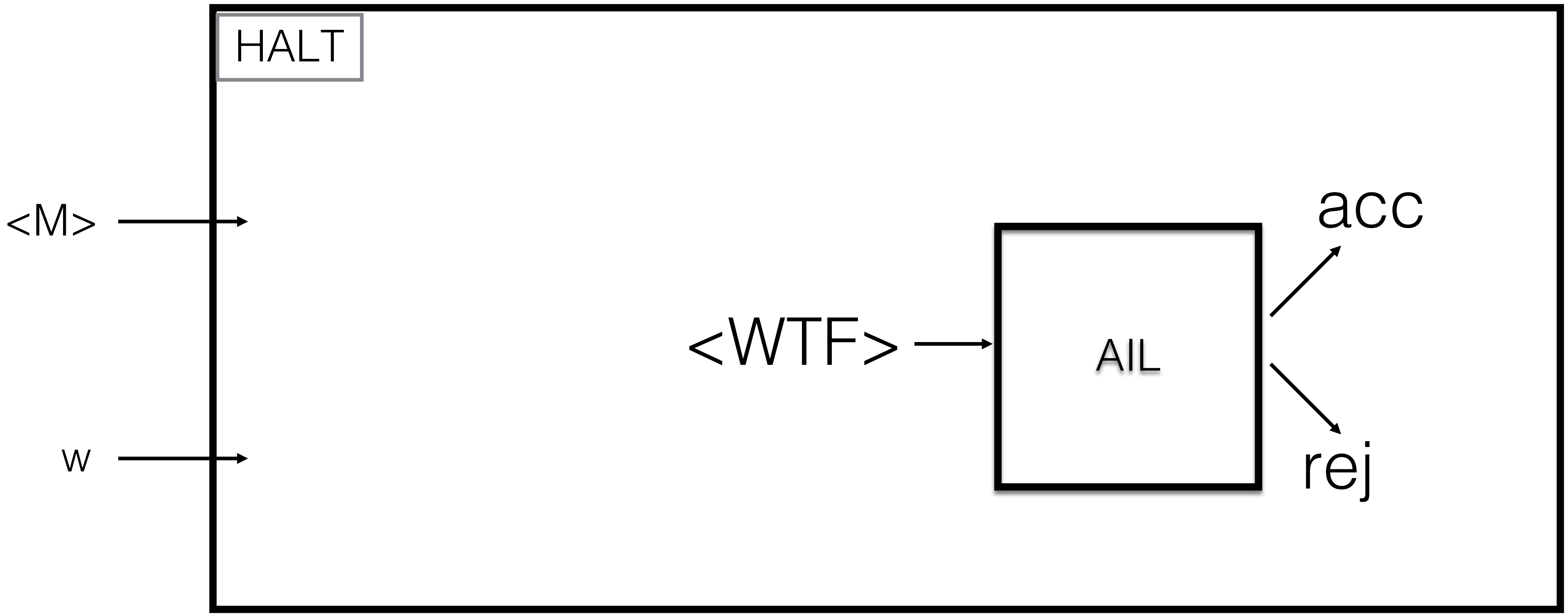
$HALT = \{ \langle M, w \rangle \mid M$ halts on $w \}$



Rice's Theorem

- ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}

HALT = { **<M,w>** | *M* halts on w }

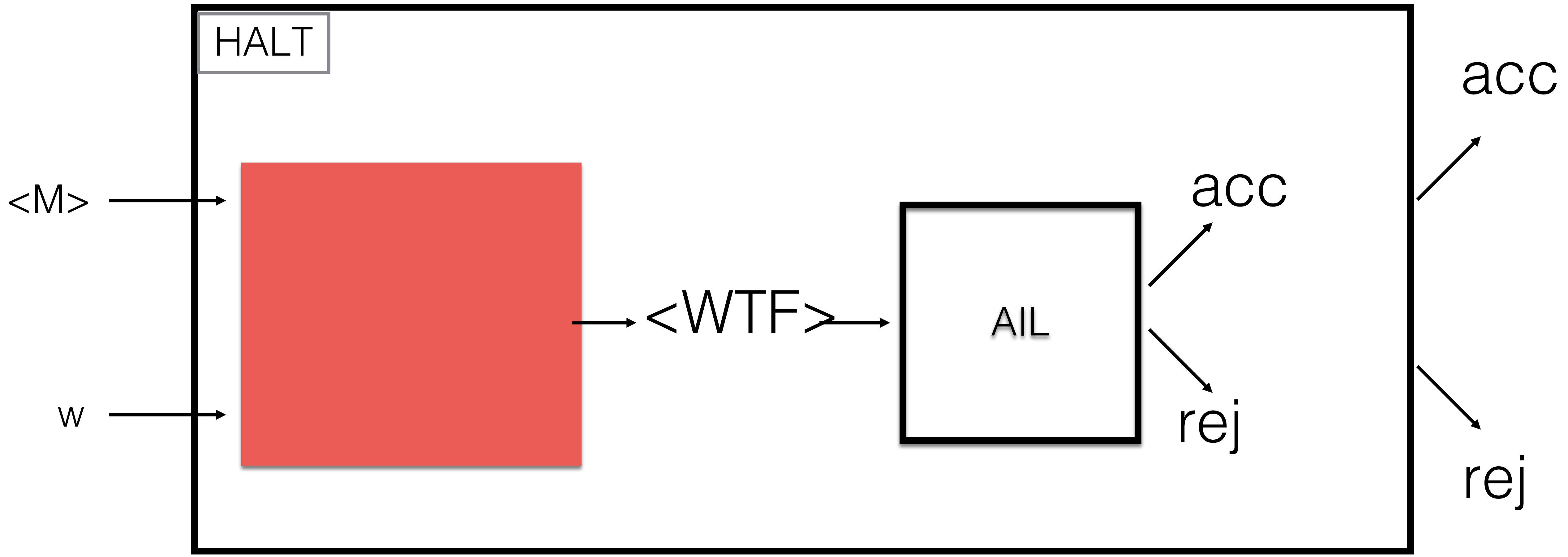


M halts on W iff ACCEPT(WTF) is in L

Rice's Theorem

- ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}

  HALT = { <M,w> | *M* halts on w }

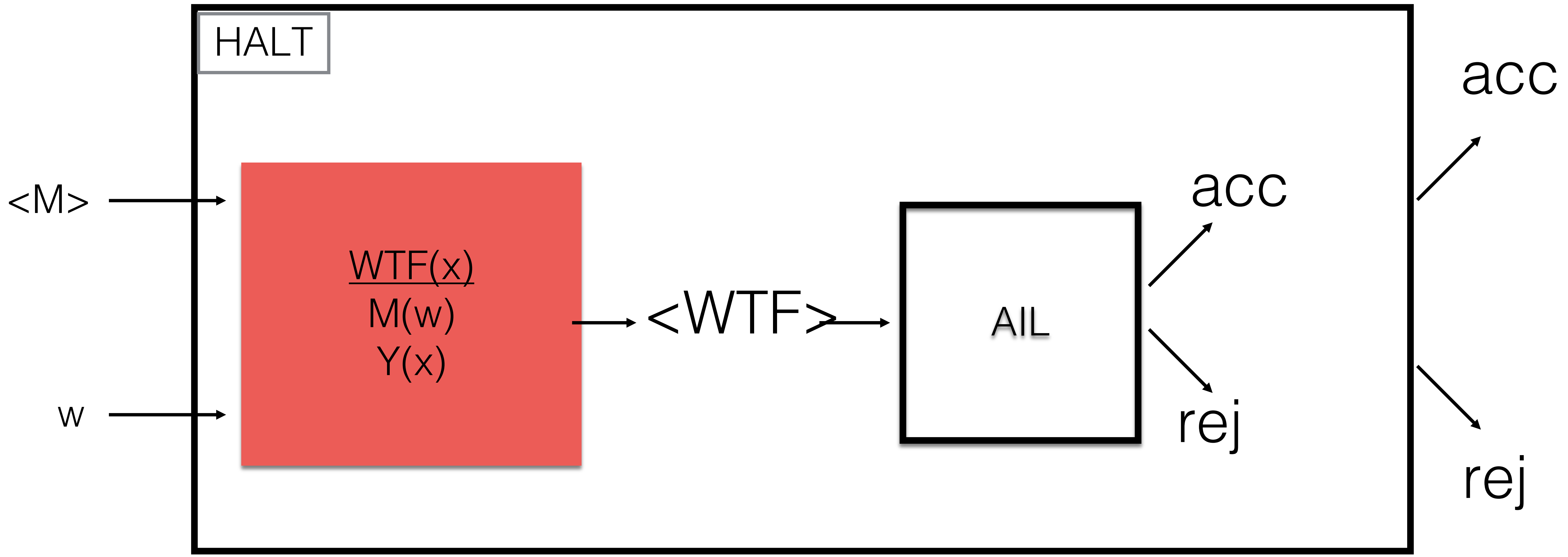


M halts on w iff ACCEPT(WTF) is in L

Rice’s Theorem

- ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}

  $HALT = \{ \langle M, w\rangle \mid M$ halts on w }

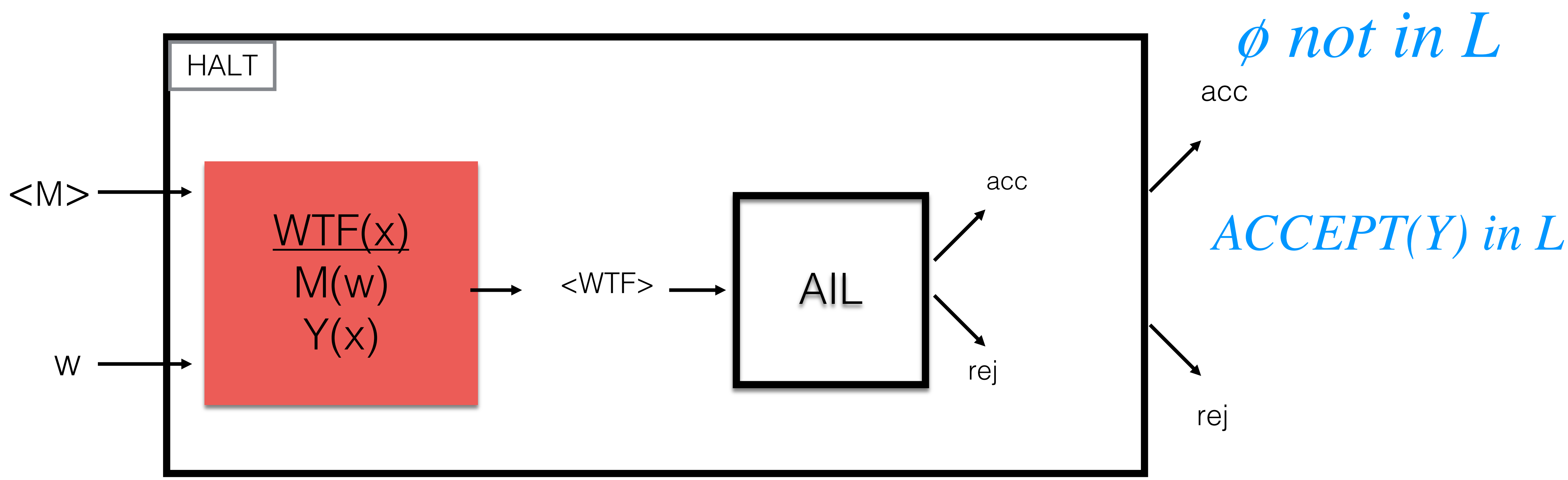


M halts on w iff ACCEPT(WTF) is in L

Assume ϕ not in L. Let Y be a TM so that ACCEPT(Y) in L

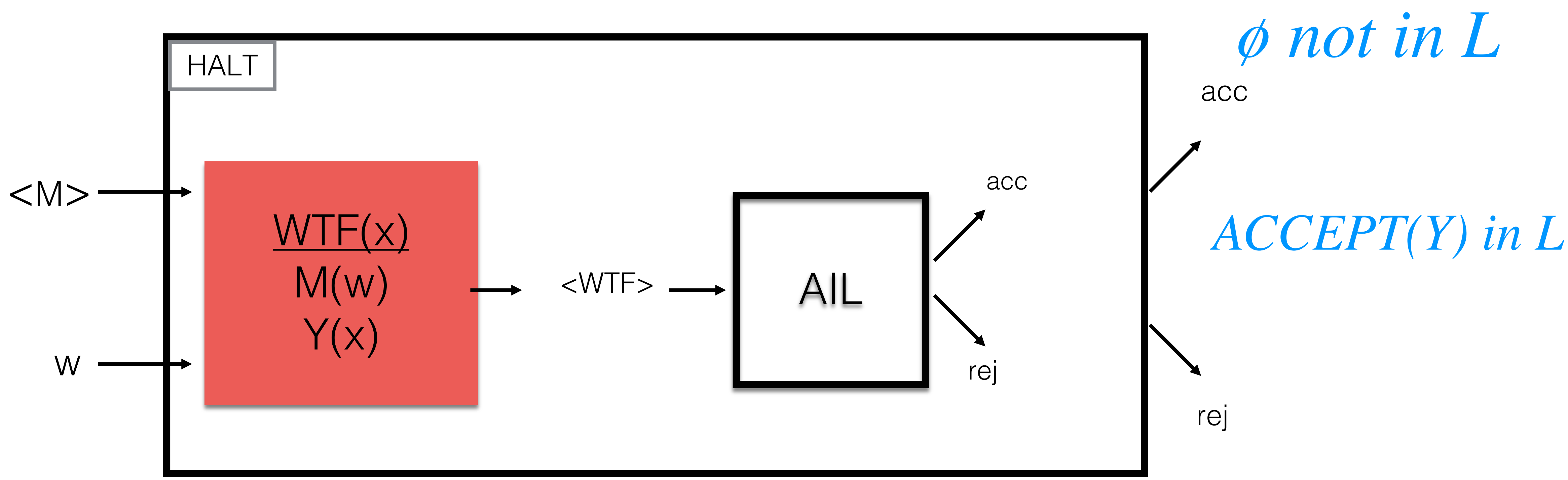
Rice's Theorem

- ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}



Rice's Theorem

- ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}



if *M halts on w then WTF(x) is Y(x) and*

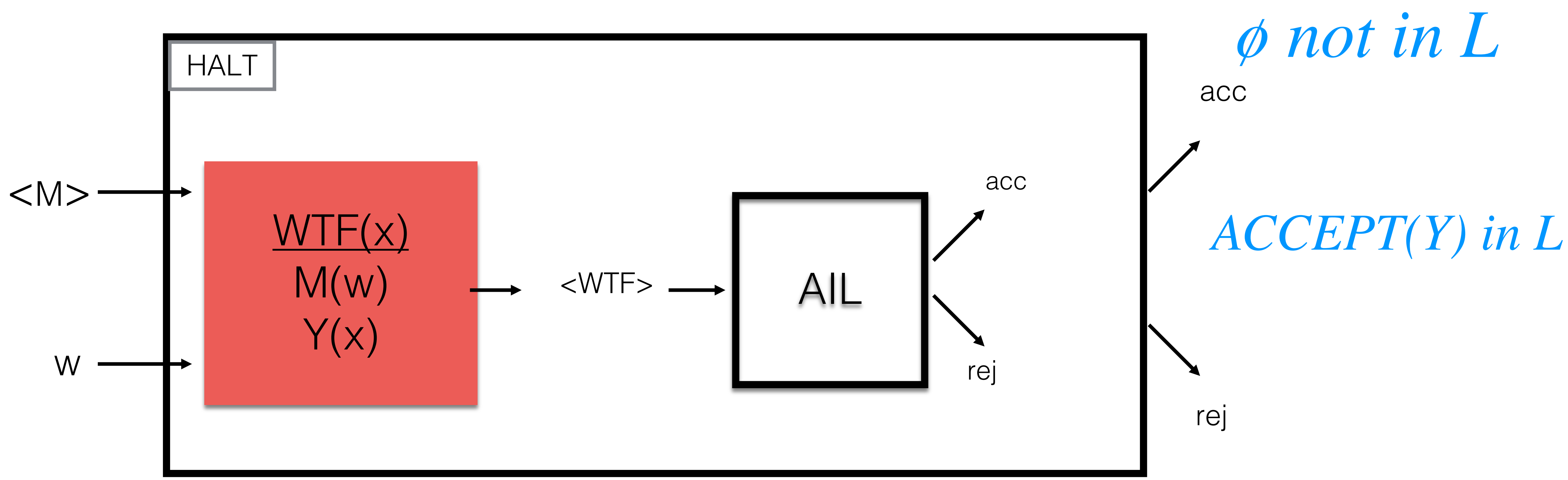
ACCEPT(WTF)=ACCEPT(Y) in L, AIL accepts

if *M doesn't halt on w then WTF(x) never halts*

so ACCEPT(WTF)=ϕ, not in L, AIL rejects

Rice's Theorem

- ACCEPTIN(L) = {<M>|ACCEPT(M) is in L}



H accepts <M,w> iff H halts on w!

contradiction

Rice's Theorem

Rice's Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that $\textsc{Accept}(Y) \in \mathcal{L}$.*
- *There is a Turing machine N such that $\textsc{Accept}(N) \notin \mathcal{L}$.*

The language $\textsc{AcceptIn}(\mathcal{L}) := \{\langle M \rangle \mid \textsc{Accept}(M) \in \mathcal{L}\}$ is undecidable.

- example: {<M>| M accepts the empty string}

Rice's Theorem

Rice's Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that* $\textsc{Accept}(Y) \in \mathcal{L}$.
- *There is a Turing machine N such that* $\textsc{Accept}(N) \notin \mathcal{L}$.

The language $\textsc{AcceptIn}(\mathcal{L}) := \{\langle M \rangle \mid \textsc{Accept}(M) \in \mathcal{L}\}$ *is undecidable.*

- example: {<M>| M accepts the empty string}

Let L be the set of all languages that contain the empty string.

Then AcceptIn(L) = { ⟨*M*⟩ | *M* accepts given an empty initial tape}.

Rice's Theorem

Rice's Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that* $\text{Accept}(Y) \in \mathcal{L}$.
- *There is a Turing machine N such that* $\text{Accept}(N) \notin \mathcal{L}$.

The language $\text{AcceptIn}(\mathcal{L}) := \{\langle M \rangle \mid \text{Accept}(M) \in \mathcal{L}\}$ *is undecidable.*

- example: {<M>| M accepts the empty string}

Let L be the set of all languages that contain the empty string.

Then AcceptIn(L) = { ⟨*M*⟩ | *M* accepts given an empty initial tape}.

- M1 accepts nothing : empty string is not in ø

- M2 accepts everything: empty string is in S*

Rice's Theorem

Rice's Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that $\textsc{Accept}(Y) \in \mathcal{L}$.*
- *There is a Turing machine N such that $\textsc{Accept}(N) \notin \mathcal{L}$.*

The language $\textsc{AcceptIn}(\mathcal{L}) := \{\langle M \rangle \mid \textsc{Accept}(M) \in \mathcal{L}\}$ *is undecidable.*

example: {<M>| M accepts regular language}

Rice's Theorem

Rice's Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that* $\text{Accept}(Y) \in \mathcal{L}$.
- *There is a Turing machine N such that* $\text{Accept}(N) \notin \mathcal{L}$.

The language $\text{AcceptIn}(\mathcal{L}) := \{\langle M\rangle \mid \text{Accept}(M) \in \mathcal{L}\}$ *is undecidable.*

- example: {<M>| M accepts the empty string}

Let L be the set of all regular languages. Then AcceptIn(L) = { ⟨*M*⟩ | *M* accepts a regular language}.

Rice's Theorem

Rice's Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that $\text{Accept}(Y) \in \mathcal{L}$.*
- *There is a Turing machine N such that $\text{Accept}(N) \notin \mathcal{L}$.*

The language $\text{AcceptIn}(\mathcal{L}) := \{\langle M \rangle \mid \text{Accept}(M) \in \mathcal{L}\}$ is undecidable.

- example: {<M>| M accepts the empty string}

Let L be the set of all regular languages. Then AcceptIn(L) = { $\langle M \rangle$ | *M* accepts a regular language}.

- M1 accepts O*

- M2 accepts $\{0^n1^n : n \geq 0\}$

Rice's Rejection Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that $\textsc{Reject}(Y) \in \mathcal{L}$*
- *There is a Turing machine N such that $\textsc{Reject}(N) \notin \mathcal{L}$.*

The language $\textsc{RejectIn}(\mathcal{L}) := \{\langle M \rangle \mid \textsc{Reject}(M) \in \mathcal{L}\}$ *is undecidable.*

Rice's Halting Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that $\textsc{Halt}(Y) \in \mathcal{L}$*
- *There is a Turing machine N such that $\textsc{Halt}(N) \notin \mathcal{L}$.*

The language $\textsc{HaltIn}(\mathcal{L}) := \{\langle M \rangle \mid \textsc{Halt}(M) \in \mathcal{L}\}$ *is undecidable.*

Rice's Divergence Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that $\textsc{Diverge}(Y) \in \mathcal{L}$*
- *There is a Turing machine N such that $\textsc{Diverge}(N) \notin \mathcal{L}$.*

The language $\textsc{DivergeIn}(\mathcal{L}) := \{\langle M \rangle \mid \textsc{Diverge}(M) \in \mathcal{L}\}$ *is undecidable.*

Rice's Decision Theorem. *Let $\mathcal{L}$ be any set of languages that satisfies the following conditions:*

- *There is a Turing machine Y such that* ***decides*** *an language in $\mathcal{L}$.*
- *There is a Turing machine N such that* ***decides*** *an language not in $\mathcal{L}$.*

The language $\textsc{DecideIn}(\mathcal{L}) := \{\langle M \rangle \mid M \textbf{ decides} \text{ a language in } \mathcal{L}\}$ *is undecidable.*

Exercise:

The language $L := \left\{\langle M, w\rangle \;\middle|\; M \text{ accepts } w^k \text{ for every integer } k \geq 0\right\}$ *is undecidable.*