Theory of Computation
Undecidable Languages

Arjun Chandrasekhar

Undecidability Proofs

We want to show that language B is undecidable

Technique: Use reducibility to prove that a language is decidable

1. AFSOC B is decidable
2. Show that $A \leq_T B$
   "If we can decide B we can also decide A"
3. But A is known to be undecidable
   - This is a contradiction!
4. We conclude that B was never decidable in the first place

The language E_{TM}

Consider the following language

$$\mathrm{E}_{\mathrm{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \emptyset\}$$

- We receive a TM description $\langle M \rangle$ as input
- We want to determine whether M is capable of accepting any strings or not
- We accept $\langle M \rangle$ if M rejects or loops on every string; otherwise we reject $\langle M \rangle$

$\mathrm{E_{TM}}$ is undecidable

Let's prove that $\mathrm{E_{TM}}$ is undecidable

$$\mathrm{E_{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \emptyset\}$$

- Hint 1: Reduce from $\mathrm{A_{TM}}$
- Hint 2: Your solution will involve constructing a machine P at runtime

$\mathrm{E_{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{E_{TM}}$ is undecidable

$$\mathrm{E_{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \emptyset\}$$

AFSOC machine M_E decides $\mathrm{E_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
    - 2.1 P receives s as input
    - 2.2 If $s = w$, run M on s
      If $s \neq w$, reject
      M and w are hard-coded constants

$\mathrm{E_{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{E_{TM}}$ is undecidable

$$\mathrm{E_{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \emptyset\}$$

AFSOC machine M_E decides $\mathrm{E_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
    2.1 P receives s as input
    2.2 If $s = w$, run M on s
    If $s \neq w$, reject
    M and w are hard-coded constants

What is $L(P)$?

If M accepts w then $L(P) = \{w\}$

If M doesn't accept w then $L(P) = \emptyset$

$\langle P \rangle \in \mathrm{E_{TM}} \Leftrightarrow \langle M, w \rangle \notin \mathrm{A_{TM}}$

E_{TM} is undecidable (approach 1)

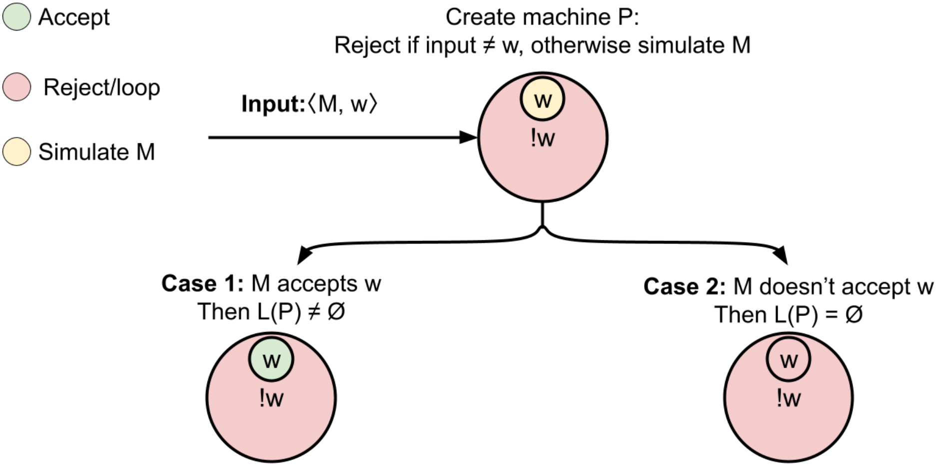
Let's prove that E_{TM} is undecidable

$$\mathrm{E}_{\mathrm{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \emptyset\}$$

AFSOC machine M_E decides E_{TM}. We will construct a machine D to decide A_{TM}

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   - 2.1 P receives s as input
   - 2.2 If $s = w$, run M on s
     If $s \neq w$, reject
     M and w are hard-coded constants
3. Use M_E to check if $\langle P \rangle \in \mathrm{E}_{\mathrm{TM}}$
   - 3.1 If M_E accepts $\langle P \rangle$, D rejects $\langle M, w \rangle$
   - 3.2 If M_E rejects $\langle P \rangle$, D accepts $\langle M, w \rangle$

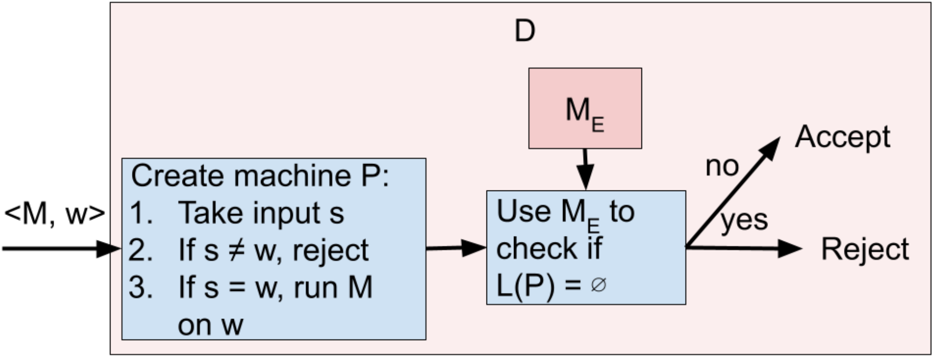
E_{TM} is undecidable (approach 1)



If we can check if L(P) = Ø, we can infer whether M accepts w

E_{TM} is undecidable (approach 1)

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ accepts } w\}$$
$$E_{TM} = \{\langle M\rangle \mid L(M) = \varnothing\}$$



If we can decide E_{TM}, we can decide A_{TM}

$\mathrm{E_{TM}}$ is undecidable (approach 2)

Let's prove that $\mathrm{E_{TM}}$ is undecidable

$$\mathrm{E_{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \emptyset\}$$

AFSOC machine M_E decides $\mathrm{E_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   - 2.1 P receives s as input
   - 2.2 Ignore s, run M on w

     M and w are hard-coded constants

What is $L(P)$?

If M accepts w then $L(P) = \Sigma^*$

If M doesn't accept w then $L(P) = \emptyset$

$\langle P \rangle \in \mathrm{E_{TM}} \Leftrightarrow \langle M, w \rangle \notin \mathrm{A_{TM}}$

E_{TM} is undecidable (approach 2)

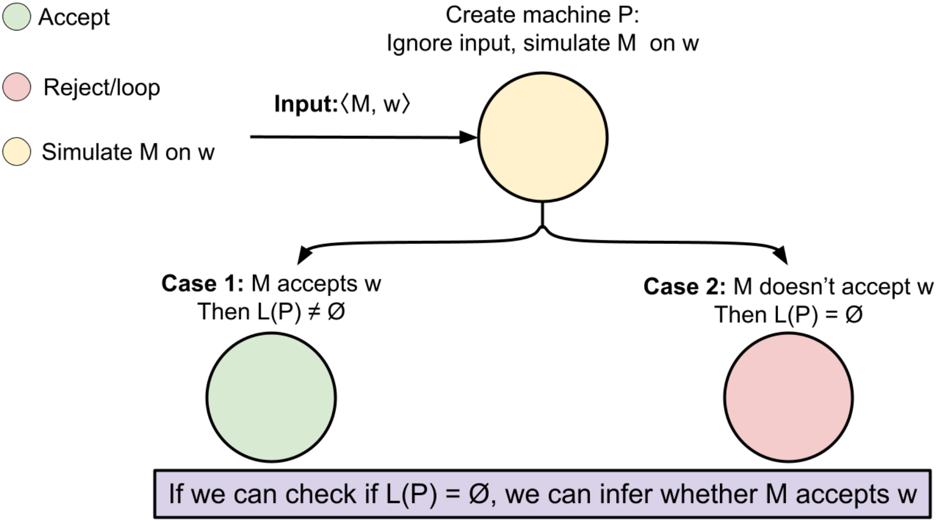
Let's prove that E_{TM} is undecidable

$$\mathrm{E}_{\mathrm{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \emptyset\}$$

AFSOC machine M_E decides E_{TM}. We will construct a machine D to decide A_{TM}

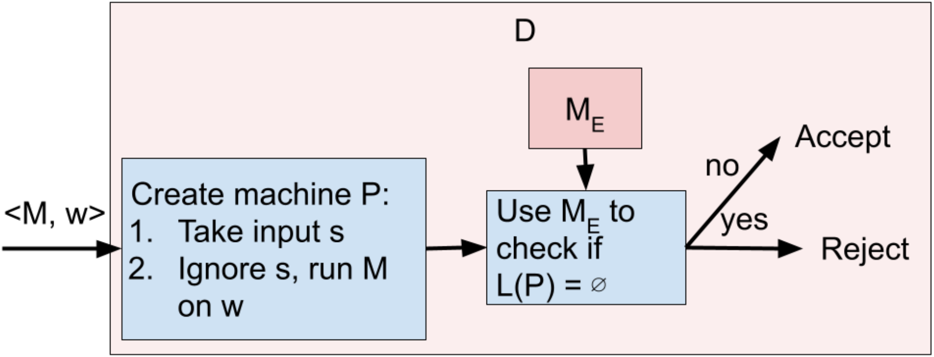
1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   - 2.1 P receives s as input
   - 2.2 Ignore s, run M on w
     M and w are hard-coded constants
3. Use M_E to check if $\langle P \rangle \in \mathrm{E}_{\mathrm{TM}}$
   - 3.1 If M_E accepts $\langle P \rangle$, D rejects $\langle M, w \rangle$
   - 3.2 If M_E rejects $\langle P \rangle$, D accepts $\langle M, w \rangle$

E_{TM} is undecidable (approach 2)



E_{TM} is undecidable (approach 2)

A_{TM} = {<M, w> | M accepts w}
E_{TM} = {<M> | L(M) = ∅}



If we can decide E_{TM}, we can decide A_{TM}

The language $\mathrm{ALL}_{\mathrm{TM}}$

Consider the following language

$$\mathrm{ALL}_{\mathrm{TM}} = \{\langle M \rangle | L(M) = \Sigma^*\}$$

We receive a TM description as input, and want to figure out if that TM accepts everything

$\mathrm{ALL_{TM}}$ is undecidable

Let's prove that $\mathrm{ALL_{TM}}$ is undecidable

$\mathrm{ALL_{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \Sigma^*\}$

- Hint 1: Reduce from $\mathrm{A_{TM}}$
- Hint 2: Your solution will involve constructing a machine P at runtime

$\mathrm{ALL_{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{ALL_{TM}}$ is undecidable

$\mathrm{ALL_{TM}} = \{\langle M \rangle | M$ is a Turing Machine, $L(M) = \Sigma^*\}$

AFSOC machine M_A decides $\mathrm{ALL_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   2.1 P receives s as input
   2.2 If $s = w$, run M on s
   If $s \neq w$, accept
   M and w are hard-coded constants

$\mathrm{ALL}_{\mathrm{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{ALL}_{\mathrm{TM}}$ is undecidable

$\mathrm{ALL}_{\mathrm{TM}} = \{\langle M \rangle | M$ is a Turing Machine, $L(M) = \Sigma^*\}$

AFSOC machine M_A decides $\mathrm{ALL}_{\mathrm{TM}}$. We will construct a machine D to decide A_{TM}

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   2.1 P receives s as input
   2.2 If $s = w$, run M on s
   If $s \neq w$, accept
   M and w are hard-coded constants

What is $L(P)$?

If M accepts w then $L(P) = \Sigma^*$

If M doesn't accept w then $L(P) = \Sigma^* \setminus \{w\}$

$\langle P \rangle \in \mathrm{ALL}_{\mathrm{TM}} \Leftrightarrow \langle M, w \rangle \in \mathrm{A}_{\mathrm{TM}}$

$\mathrm{ALL_{TM}}$ is undecidable (approach 1)

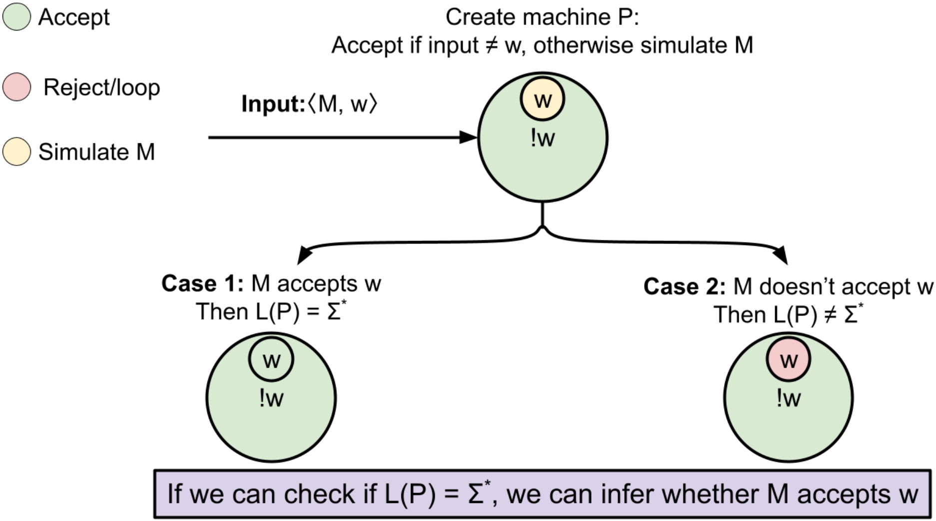
Let's prove that $\mathrm{ALL_{TM}}$ is undecidable

$\mathrm{ALL_{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \Sigma^*\}$

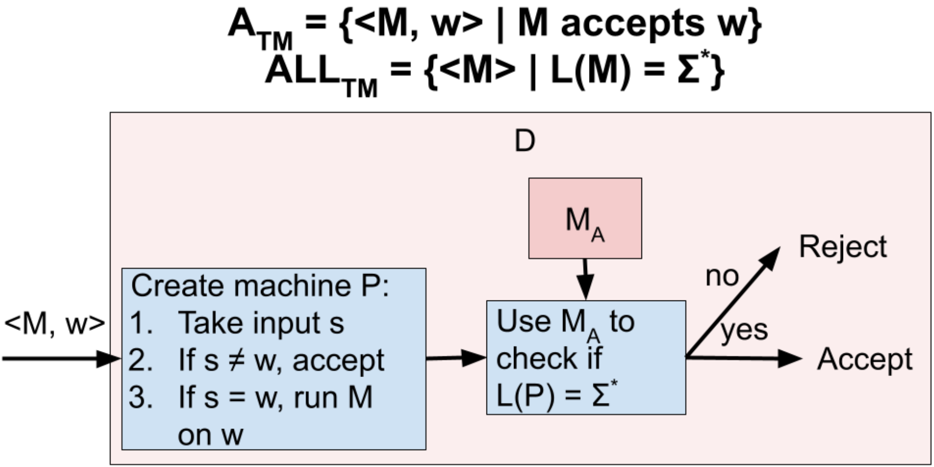
AFSOC machine M_A decides $\mathrm{ALL_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   - 2.1 P receives s as input
   - 2.2 If $s = w$, run M on s
     If $s \neq w$, accept
     M and w are hard-coded constants
3. Use M_A to check if $\langle P \rangle \in \mathrm{ALL_{TM}}$
   - 3.1 If M_A accepts $\langle P \rangle$, D accepts $\langle M, w \rangle$
   - 3.2 If M_A rejects $\langle P \rangle$, D rejects $\langle M, w \rangle$

$\mathrm{ALL}_{\mathrm{TM}}$ is undecidable (approach 1)



$\mathrm{ALL_{TM}}$ is undecidable (approach 1)



If we can decide $\mathrm{ALL_{TM}}$, we can decide $\mathrm{A_{TM}}$

$\mathrm{ALL_{TM}}$ is undecidable (approach 2)

Let's prove that $\mathrm{ALL_{TM}}$ is undecidable

$\mathrm{ALL_{TM}} = \{\langle M\rangle | M \text{ is a Turing Machine, } L(M) = \Sigma^*\}$

AFSOC machine M_A decides $\mathrm{ALL_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w\rangle$ as input
2. Create a new machine P
    2.1 P receives s as input
    2.2 Ignore s, run M on w
    M and *w* are hard-coded constants

$\mathrm{ALL_{TM}}$ is undecidable (approach 2)

Let's prove that $\mathrm{ALL_{TM}}$ is undecidable

$\mathrm{ALL_{TM}} = \{\langle M \rangle | M$ is a Turing Machine, $L(M) = \Sigma^*\}$

AFSOC machine M_A decides $\mathrm{ALL_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   - 2.1 P receives s as input
   - 2.2 Ignore s, run M on w

     M and w are hard-coded constants

What is $L(P)$?

If M accepts w then $L(P) = \Sigma^*$

If M doesn't accept w then $L(P) = \emptyset$

$\langle P \rangle \in \mathrm{ALL_{TM}} \Leftrightarrow \langle M, w \rangle \in \mathrm{A_{TM}}$

$\mathrm{ALL}_{\mathrm{TM}}$ is undecidable (approach 2)

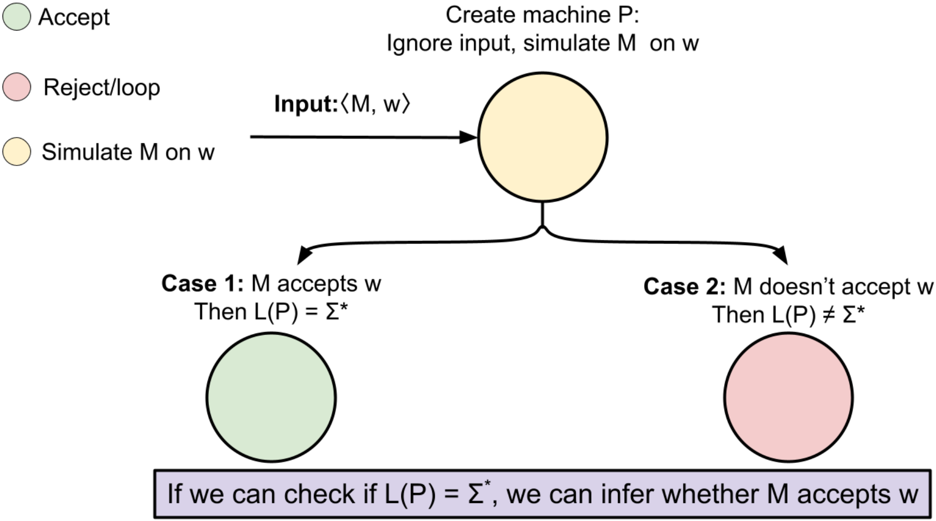
Let's prove that $\mathrm{ALL}_{\mathrm{TM}}$ is undecidable

$\mathrm{ALL}_{\mathrm{TM}} = \{\langle M \rangle | M \text{ is a Turing Machine, } L(M) = \Sigma^*\}$

AFSOC machine M_A decides $\mathrm{ALL}_{\mathrm{TM}}$. We will construct a machine D to decide A_{TM}

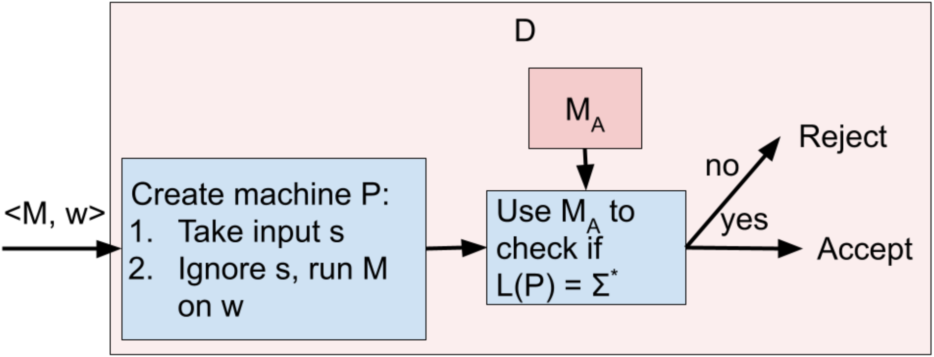
1. D receives $\langle M, w \rangle$ as input
2. Create a new machine P
   2.1 P receives s as input
   2.2 Ignore s, run M on w
       M and w are hard-coded constants
3. Use M_A to check if $\langle P \rangle \in \mathrm{ALL}_{\mathrm{TM}}$
   3.1 If M_A accepts $\langle P \rangle$, D accepts $\langle M, w \rangle$
   3.2 If M_A rejects $\langle P \rangle$, D rejects $\langle M, w \rangle$

$\mathrm{ALL}_{\mathrm{TM}}$ is undecidable (approach 2)



$\mathrm{ALL_{TM}}$ is undecidable (approach 2)

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ accepts } w\}$$
$$ALL_{TM} = \{\langle M\rangle \mid L(M) = \Sigma^*\}$$



If we can decide ALL_{TM}, we can decide A_{TM}

The language $\mathrm{EQ}_{\mathrm{TM}}$

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) = L(M_2)\}$$

We receive two Turing machine descriptions, and we want to determine out if the two machines are equivalent

- Can we write a script to check that your programming assignment submissions are equivalent to my solution code?
  - "equivalent" as in "the EXACT same output on ALL (possible) test cases"

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$$

We will reduce from each of the following languages

$$\begin{aligned} \mathrm{A}_{\mathrm{TM}} &= \{\langle M, w\rangle | w \in L(M)\} \\ \mathrm{E}_{\mathrm{TM}} &= \{\langle M\rangle | L(M) = \emptyset\} \\ \mathrm{ALL}_{\mathrm{TM}} &= \{\langle M\rangle | L(M) = \Sigma^*\} \end{aligned}$$

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) = L(M_2)\}$$

Reduce from A_{TM}: AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide A_{TM}

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine M_2
   2.1 M_2 receives s as input
   2.2 If $s = w$, M_2 accepts. Otherwise, M_2 runs M on s

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$$

Reduce from A_{TM}: AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide A_{TM}

1. D receives $\langle M, w\rangle$ as input
2. Create a new machine M_2
   - 2.1 M_2 receives s as input
   - 2.2 If $s = w$, M_2 accepts. Otherwise, M_2 runs M on s

When are M and M_2 equivalent?

$L(M) = L(M_2) \Leftrightarrow M$ accepts w

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$$

Reduce from A_{TM}**:** AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide A_{TM}

1. D receives $\langle M, w\rangle$ as input
2. Create a new machine M_2
   2.1 M_2 receives s as input
   2.2 If $s = w$, M_2 accepts. Otherwise, M_2 runs M on s
3. Use M_{EQ} to check if $\langle M, M_2\rangle \in \mathrm{EQ}_{\mathrm{TM}}$

$\mathrm{EQ_{TM}}$ is undecidable (approach 1)

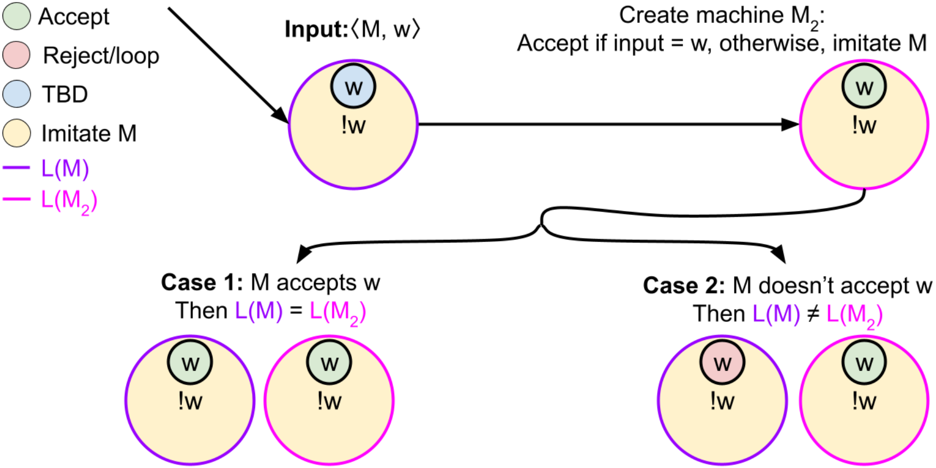
Let's prove that $\mathrm{EQ_{TM}}$ is undecidable

$$\mathrm{EQ_{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) = L(M_2)\}$$

Reduce from $\mathrm{A_{TM}}$: AFSOC machine M_{EQ} decides $\mathrm{EQ_{TM}}$. We will construct a machine D to decide $\mathrm{A_{TM}}$

1. D receives $\langle M, w \rangle$ as input
2. Create a new machine M_2
   - 2.1 M_2 receives s as input
   - 2.2 If $s = w$, M_2 accepts. Otherwise, M_2 runs M on s
3. Use M_{EQ} to check if $\langle M, M_2 \rangle \in \mathrm{EQ_{TM}}$
   - 3.1 If M_{EQ} accepts $\langle M, M_2 \rangle$, then D accepts $\langle M, w \rangle$
   - 3.2 Otherwise D rejects $\langle M, w \rangle$

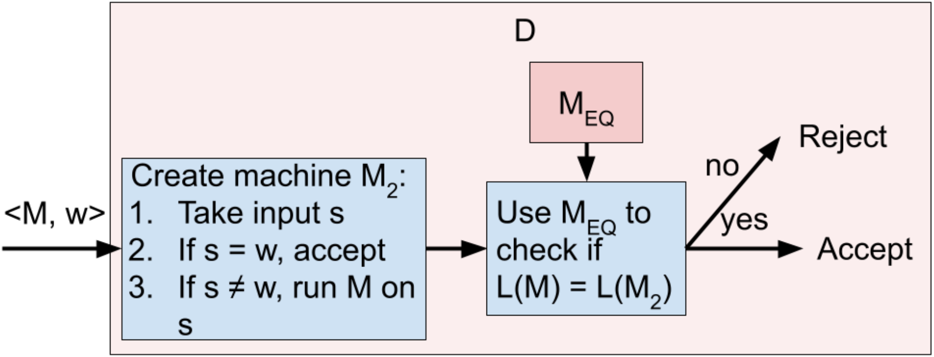
$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 1)



If we can check if $L(M) = L(M_2)$, we can infer whether M accepts w

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 1)

A_{TM} = {<M, w> | M accepts w}
EQ_{TM} = {<M_1, M_2> | $L(M_1) = L(M_2)$}



If we can decide EQ_{TM}, we can decide A_{TM}

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 2)

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$$

Reduce from E_{TM}**:** AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide E_{TM}

1. D receives $\langle M \rangle$ as input
2. Create a new machine M_2 that recognizes $\emptyset$

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 2)

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) = L(M_2)\}$$

Reduce from E_{TM}: AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide E_{TM}

1. D receives $\langle M \rangle$ as input
2. Create a new machine M_2 that recognizes $\emptyset$

When are M and M_2 equivalent?
$L(M) = L(M_2) \Leftrightarrow L(M) = \emptyset$

$\mathrm{EQ_{TM}}$ is undecidable (approach 2)

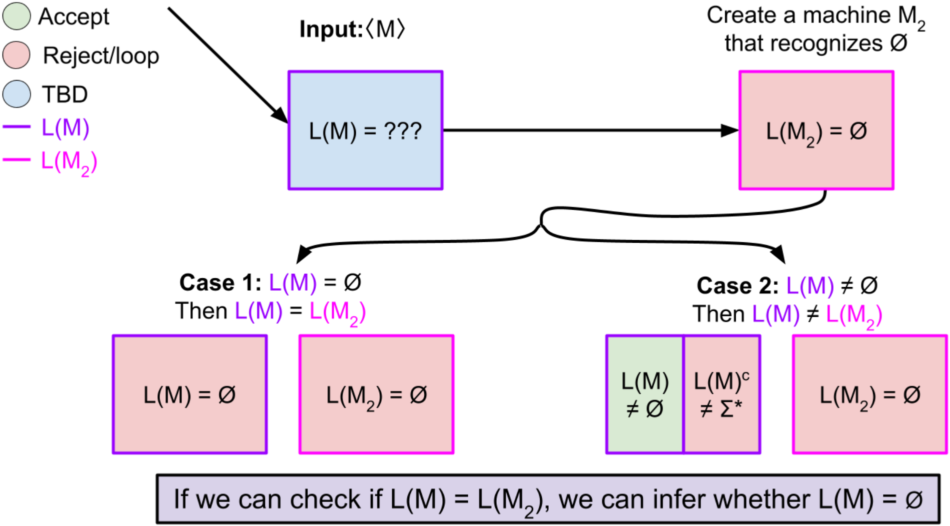
Let's prove that $\mathrm{EQ_{TM}}$ is undecidable

$$\mathrm{EQ_{TM}} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$$

Reduce from $\mathrm{E_{TM}}$: AFSOC machine M_{EQ} decides $\mathrm{EQ_{TM}}$. We will construct a machine D to decide $\mathrm{E_{TM}}$

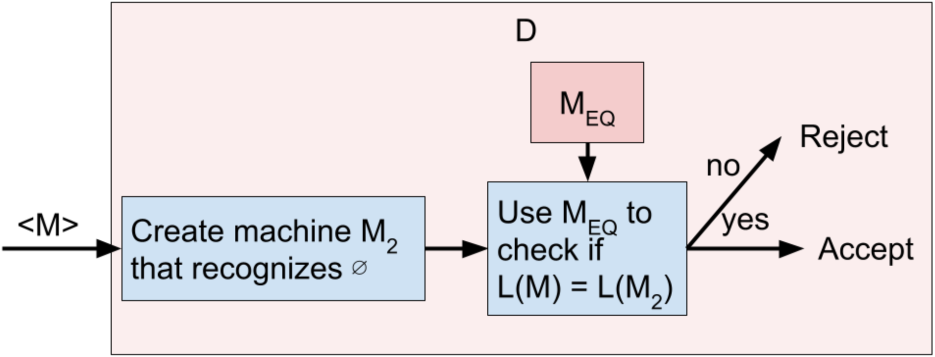
1. D receives $\langle M\rangle$ as input
2. Create a new machine M_2 that recognizes $\emptyset$
3. Use M_{EQ} to check if $\langle M, M_2\rangle \in \mathrm{EQ_{TM}}$
   3.1 If M_{EQ} accepts $\langle M, M_2\rangle$, then D accepts $\langle M\rangle$
   3.2 Otherwise D rejects $\langle M\rangle$

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 2)



$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 2)

$$E_{TM} = \{<M> \mid L(M) = \varnothing\}$$
$$EQ_{TM} = \{<M_1, M_2> \mid L(M_1) = L(M_2)\}$$



If we can decide EQ_{TM}, we can decide E_{TM}

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 3)

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) = L(M_2)\}$$

Reduce from $\mathrm{ALL}_{\mathrm{TM}}$**:** AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide $\mathrm{ALL}_{\mathrm{TM}}$

1. D receives $\langle M \rangle$ as input
2. Create a new machine M_2 that recognizes Σ^*

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 3)

Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$$

Reduce from $\mathrm{ALL}_{\mathrm{TM}}$**:** AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide $\mathrm{ALL}_{\mathrm{TM}}$

1. D receives $\langle M \rangle$ as input
2. Create a new machine M_2 that recognizes Σ^*

When are M and M_2 equivalent?
$L(M) = L(M_2) \Leftrightarrow L(M) = \Sigma^*$

$\mathrm{EQ}_{\mathrm{TM}}$ is undecidable (approach 3)

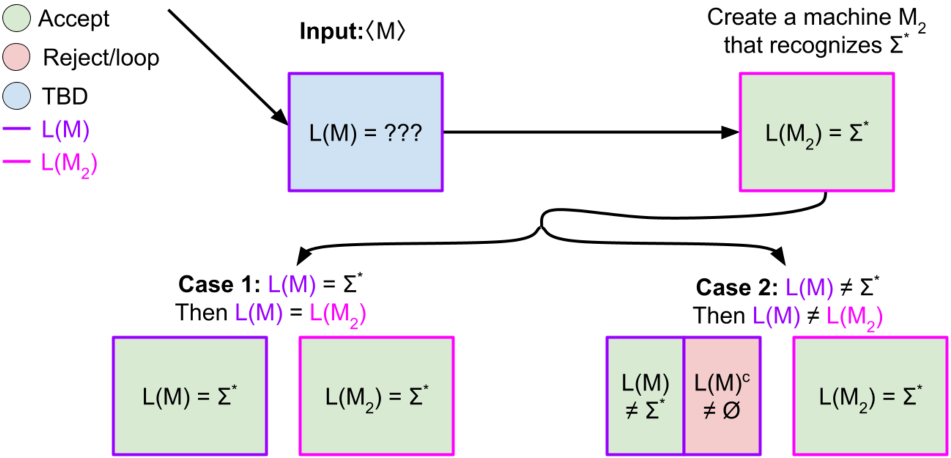
Let's prove that $\mathrm{EQ}_{\mathrm{TM}}$ is undecidable

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$$

Reduce from $\mathrm{ALL}_{\mathrm{TM}}$**:** AFSOC machine M_{EQ} decides $\mathrm{EQ}_{\mathrm{TM}}$. We will construct a machine D to decide $\mathrm{ALL}_{\mathrm{TM}}$

1. D receives $\langle M\rangle$ as input
2. Create a new machine M_2 that recognizes Σ^*
3. Use M_{EQ} to check if $\langle M, M_2\rangle \in \mathrm{EQ}_{\mathrm{TM}}$
   3.1 If M_{EQ} accepts $\langle M, M_2\rangle$, then D accepts $\langle M\rangle$
   3.2 Otherwise D rejects $\langle M\rangle$

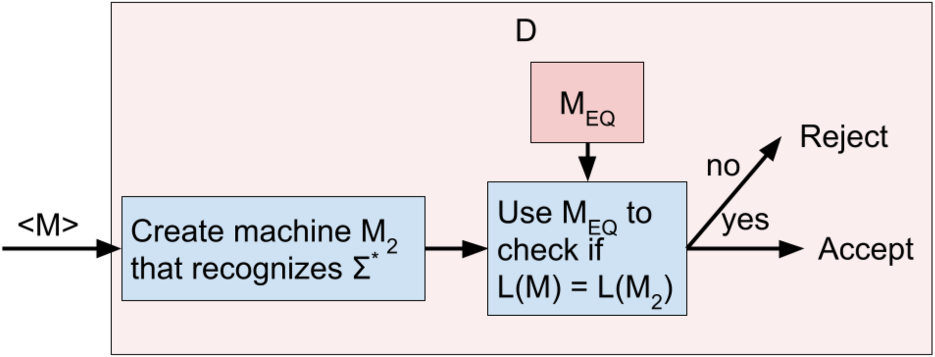
EQ_{TM} is undecidable (approach 3)



If we can check if $L(M) = L(M_2)$, we can infer whether $L(M) = \Sigma^*$

EQ_{TM} is undecidable (approach 3)

$$ALL_{TM} = \{\langle M \rangle \mid L(M) = \Sigma^*\}$$
$$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid L(M_1) = L(M_2)\}$$



If we can decide EQ_{TM}, we can decide ALL_{TM}

The language $\mathrm{SUB}_{\mathrm{TM}}$

Consider the following language

$$\mathrm{SUB}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) \subseteq L(M_2)\}$$

We receive two machines M_1, M_2 as input. We want to determine if M_1 is contained within M_2

$\mathrm{SUB_{TM}}$ is undecidable

Let's prove that $\mathrm{SUB_{TM}}$ is undecidable

$$\mathrm{SUB_{TM}} = \{\langle M_1, M_2\rangle | L(M_1) \subseteq L(M_2)\}$$

We will reduce from each of the following languages

$$\begin{aligned}
\mathrm{E_{TM}} &= \{\langle M\rangle | L(M) = \emptyset\} \\
\mathrm{ALL_{TM}} &= \{\langle M\rangle | L(M) = \Sigma^*\} \\
\mathrm{EQ_{TM}} &= \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}
\end{aligned}$$

$\mathrm{SUB}_{\mathrm{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{SUB}_{\mathrm{TM}}$ is undecidable

$$\mathrm{SUB}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from E_{TM}: AFSOC $\mathrm{SUB}_{\mathrm{TM}}$ is decided by machine M_S. We will construct a machine D to decide E_{TM} as follows:

1. D takes $\langle M\rangle$ as input
2. Construct a machine M_2 that recognizes $\emptyset$

$\mathrm{SUB_{TM}}$ is undecidable (approach 1)

Let's prove that $\mathrm{SUB_{TM}}$ is undecidable

$$\mathrm{SUB_{TM}} = \{\langle M_1, M_2\rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{E_{TM}}$: AFSOC $\mathrm{SUB_{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{E_{TM}}$ as follows:

1. D takes $\langle M\rangle$ as input
2. Construct a machine M_2 that recognizes $\emptyset$

When does M_2 contain M?

$L(M) \subseteq L(M_2) \Leftrightarrow L(M) \subseteq \emptyset \Leftrightarrow L(M) = \emptyset$

$\langle M, M_2\rangle \in \mathrm{SUB_{TM}} \Leftrightarrow \langle M\rangle \in \mathrm{E_{TM}}$

$\mathrm{SUB_{TM}}$ is undecidable (approach 1)

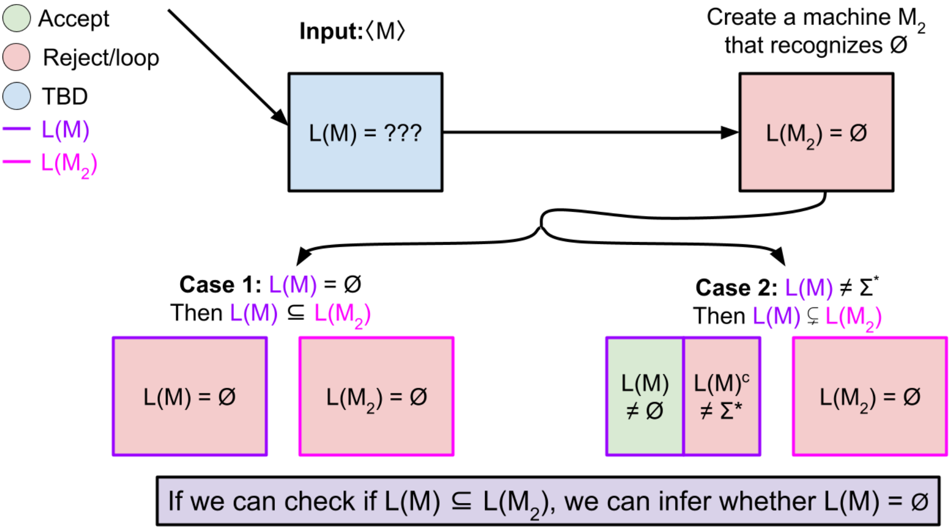
Let's prove that $\mathrm{SUB_{TM}}$ is undecidable

$$\mathrm{SUB_{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{E_{TM}}$: AFSOC $\mathrm{SUB_{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{E_{TM}}$ as follows:

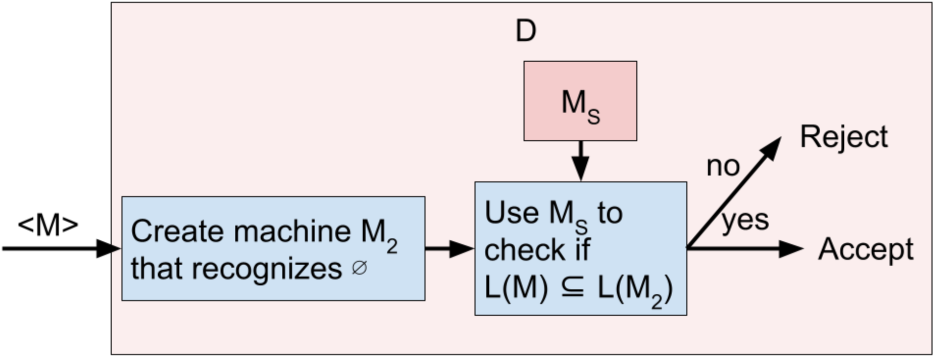
1. D takes $\langle M \rangle$ as input
2. Construct a machine M_2 that recognizes $\emptyset$
3. Use M_S to check if $\langle M, M_2 \rangle \in \mathrm{SUB_{TM}}$
   "Is M contained within a machine that accepts nothing?"
   3.1 If M_S accepts $\langle M, M_2 \rangle$, then D accepts $\langle M \rangle$
   3.2 Otherwise, D rejects $\langle M \rangle$

$\mathrm{SUB}_{\mathrm{TM}}$ is undecidable (approach 1)



$\mathrm{SUB}_{\mathrm{TM}}$ is undecidable (approach 1)

$$E_{TM} = \{\langle M \rangle \mid L(M) = \emptyset\}$$
$$SUB_{TM} = \{\langle M_1, M_2 \rangle \mid L(M_1) \subseteq L(M_2)\}$$



If we can decide SUB_{TM}, we can decide E_{TM}

$\mathrm{SUB_{TM}}$ is undecidable (approach 2)

Let's prove that $\mathrm{SUB_{TM}}$ is undecidable

$$\mathrm{SUB_{TM}} = \{\langle M_1, M_2\rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{ALL_{TM}}$: AFSOC $\mathrm{SUB_{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{ALL_{TM}}$ as follows:

1. D takes $\langle M\rangle$ as input
2. Construct a machine M_2 that recognizes Σ^*

$\mathrm{SUB}_{\mathrm{TM}}$ is undecidable (approach 2)

Let's prove that $\mathrm{SUB}_{\mathrm{TM}}$ is undecidable

$$\mathrm{SUB}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{ALL}_{\mathrm{TM}}$**:** AFSOC $\mathrm{SUB}_{\mathrm{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{ALL}_{\mathrm{TM}}$ as follows:

1. D takes $\langle M\rangle$ as input
2. Construct a machine M_2 that recognizes Σ^*

When does M contain M_2?

$L(M_2) \subseteq L(M) \Leftrightarrow \Sigma^* \subseteq L(M) \Leftrightarrow L(M) = \Sigma^*$

$\langle M_2, M\rangle \in \mathrm{SUB}_{\mathrm{TM}} \Leftrightarrow \langle M\rangle \in \mathrm{ALL}_{\mathrm{TM}}$

$\mathrm{SUB_{TM}}$ is undecidable (approach 2)

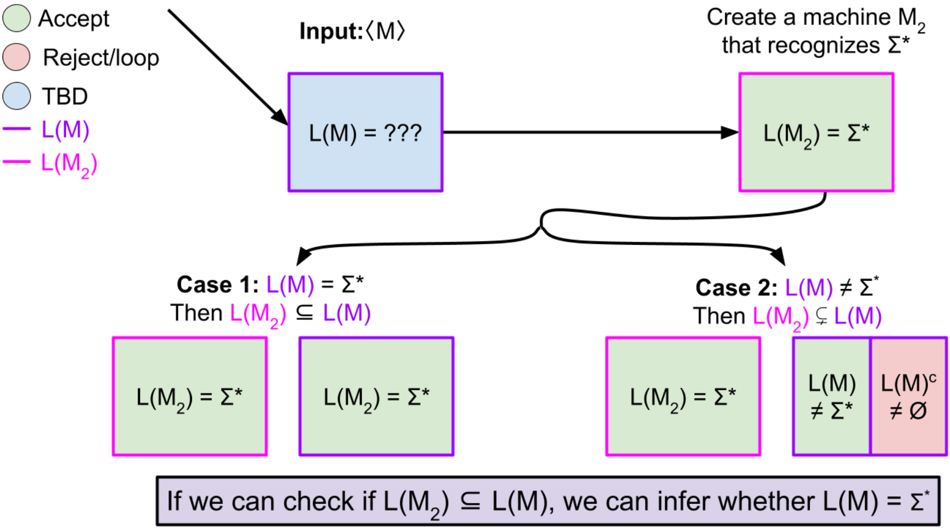
Let's prove that $\mathrm{SUB_{TM}}$ is undecidable

$$\mathrm{SUB_{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{ALL_{TM}}$: AFSOC $\mathrm{SUB_{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{ALL_{TM}}$ as follows:

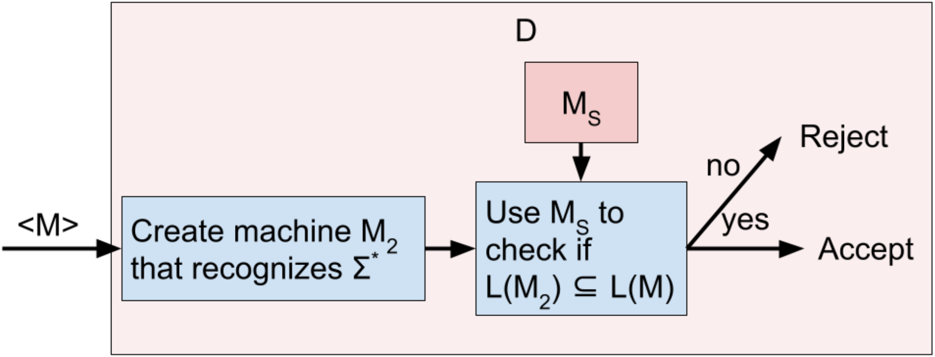
1. D takes $\langle M \rangle$ as input
2. Construct a machine M_2 that recognizes Σ^*
3. Use M_S to check if $\langle M_2, M \rangle \in \mathrm{SUB_{TM}}$
   "Does M contain a machine that accepts everything?"
   3.1 If M_S accepts $\langle M, M_2 \rangle$, then D accepts $\langle M \rangle$
   3.2 Otherwise, D rejects $\langle M \rangle$

$\mathrm{SUB_{TM}}$ is undecidable (approach 2)



$\mathrm{SUB_{TM}}$ is undecidable (approach 2)

$$ALL_{TM} = \{\langle M, w\rangle \mid L(M) = \Sigma^*\}$$
$$SUB_{TM} = \{\langle M_1, M_2\rangle \mid L(M_1) \subseteq L(M_2)\}$$



If we can decide SUB_{TM}, we can decide ALL_{TM}

$\mathrm{SUB}_{\mathrm{TM}}$ is undecidable (approach 3)

Let's prove that $\mathrm{SUB}_{\mathrm{TM}}$ is undecidable

$$\mathrm{SUB}_{\mathrm{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{EQ}_{\mathrm{TM}}$: AFSOC $\mathrm{SUB}_{\mathrm{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{EQ}_{\mathrm{TM}}$ as follows:

1. D takes $\langle M_1, M_2 \rangle$ as input

$\mathrm{SUB}_{\mathrm{TM}}$ is undecidable (approach 3)

Let's prove that $\mathrm{SUB}_{\mathrm{TM}}$ is undecidable

$$\mathrm{SUB}_{\mathrm{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{EQ}_{\mathrm{TM}}$**:** AFSOC $\mathrm{SUB}_{\mathrm{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{EQ}_{\mathrm{TM}}$ as follows:

1. D takes $\langle M_1, M_2 \rangle$ as input

When does M_1 equal M_2?

$L(M_1) = L(M_2) \Leftrightarrow L(M_1) \subseteq L(M_2) \wedge L(M_2) \subseteq L(M_1)$

$\langle M_1, M_2 \rangle \in \mathrm{EQ}_{\mathrm{TM}} \Leftrightarrow \langle M_1, M_2 \rangle, \langle M_2, M_1 \rangle \in \mathrm{SUB}_{\mathrm{TM}}$

$\mathrm{SUB}_{\mathrm{TM}}$ is undecidable (approach 3)

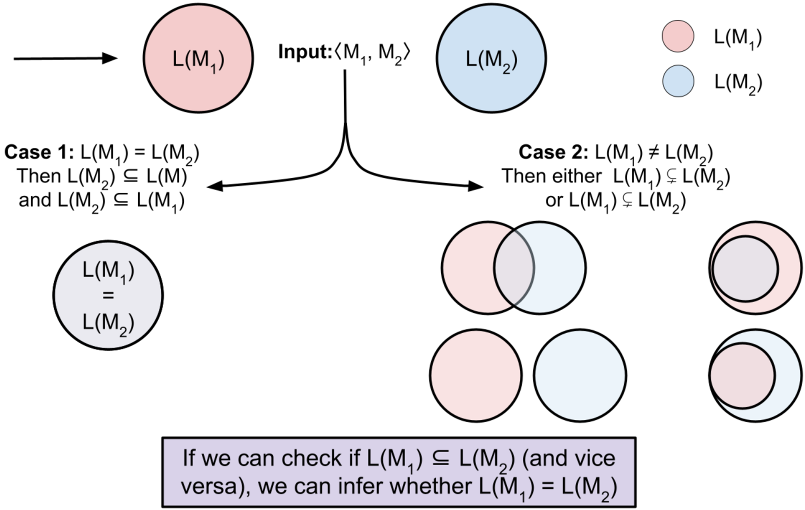
Let's prove that $\mathrm{SUB}_{\mathrm{TM}}$ is undecidable

$$\mathrm{SUB}_{\mathrm{TM}} = \{\langle M_1, M_2\rangle | L(M_1) \subseteq L(M_2)\}$$

Reduce from $\mathrm{EQ}_{\mathrm{TM}}$: AFSOC $\mathrm{SUB}_{\mathrm{TM}}$ is decided by machine M_S. We will construct a machine D to decide $\mathrm{EQ}_{\mathrm{TM}}$ as follows:

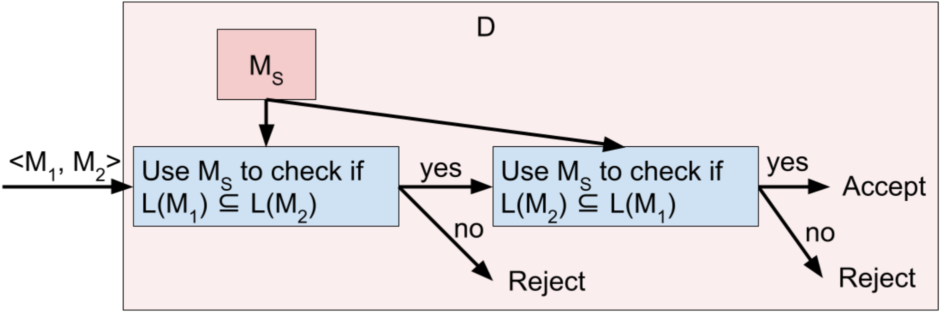
1. D takes $\langle M_1, M_2\rangle$ as input
2. Use M_S to check if $\langle M_1, M_2\rangle \in \mathrm{SUB}_{\mathrm{TM}}$ and $\langle M_2, M_1\rangle \in \mathrm{SUB}_{\mathrm{TM}}$
   "Do M_1 and M_2 contain each other?"
   2.1 If M_S accepts $\langle M_1, M_2\rangle$ and $\langle M_2, M_1\rangle$, then D accepts $\langle M_1, M_2\rangle$
   2.2 Otherwise, D rejects $\langle M_1, M_2\rangle$

$\mathrm{SUB_{TM}}$ is undecidable (approach 3)



$\mathrm{SUB_{TM}}$ is undecidable (approach 3)

$$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid L(M_1) = L(M_2)\}$$
$$SUB_{TM} = \{\langle M_1, M_2 \rangle \mid L(M_1) \subseteq L(M_2)\}$$



If we can decide SUB_{TM}, we can decide EQ_{TM}

Reducibility

Recap: If we could solve certain problems, we would be able to solve other problems

- We can use reducibility to prove undecidability
- If $A \leq_T B$ and A is known to be undecidable, then B must also be undecidable