Undecidable Problems
(unsolvable problems)
Decidable Languages

Recall that:

A language $A$ is decidable, if there is a Turing machine $M$ (decider) that accepts the language $A$ and halts on every input string.

```
Input string

Turing Machine M

Decision On Halt:
YES -> Accept
NO  -> Reject
```

```
Decider for A
```
A computational problem is decidable if the corresponding language is decidable.

We also say that the problem is solvable.
Problem: Does DFA $M$ accept the empty language $L(M) = \emptyset$?

Corresponding Language: (Decidable)

$\textit{EMPTY}_{DFA} = \{ \langle M \rangle : M \text{ is a DFA that accepts empty language } \emptyset \}$

Description of DFA $M$ as a string
(For example, we can represent $M$ as a binary string, as we did for Turing machines)
Decider for $\text{EMPTY}_{\text{DFA}}$:

On input $\langle M \rangle$:

Determine whether there is a path from the initial state to any accepting state.

- If $L(M) \neq \emptyset$, reject $\langle M \rangle$.
- If $L(M) = \emptyset$, accept $\langle M \rangle$.

Decision: Reject $\langle M \rangle$ for $L(M) \neq \emptyset$ and Accept $\langle M \rangle$ for $L(M) = \emptyset$. 
Problem: Does DFA $M$ accept a finite language?

Corresponding Language: (Decidable)

$$\text{FINITE}_{\text{DFA}} = \{ \langle M \rangle : M \text{ is a DFA that accepts a finite language} \}$$
Decider for $\text{FINITE}_{\text{DFA}}$:

On input $\langle M \rangle$:

Check if there is a walk with a cycle from the initial state to an accepting state.

Decision: 

- Reject $\langle M \rangle$ (NO)
- Accept $\langle M \rangle$ (YES)
Problem: Does DFA $M$ accept string $w$?

Corresponding Language: (Decidable)

$$A_{\text{DFA}} = \{ \langle M, w \rangle : M \text{ is a DFA that accepts string } w \}$$
Decider for $A_{DFA}$:

On input string $\langle M, w \rangle$:

Run DFA $M$ on input string $w$

If $M$ accepts $w$

Then accept $\langle M, w \rangle$ (and halt)

Else reject $\langle M, w \rangle$ (and halt)
Problem: Do DFAs $M_1$ and $M_2$ accept the same language?

Corresponding Language: (Decidable)

$EQUAL_{DFA} =$

$\{ \langle M_1, M_2 \rangle : M_1$ and $M_2$ are DFAs that accept the same languages $\}$
Decider for $EQL_{DFA}$:

On input $\langle M_1, M_2 \rangle$:

Let $L_1$ be the language of DFA $M_1$
Let $L_2$ be the language of DFA $M_2$

Construct DFA $M$ such that:

$$L(M) = (L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)$$

(combination of DFAs)
\[(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) = \emptyset\]

and

\[L_1 \cap \overline{L_2} = \emptyset\]

and

\[\overline{L_1} \cap L_2 = \emptyset\]

\[L_1 \subseteq L_2\]

\[L_2 \subseteq L_1\]

\[L_1 = L_2\]
\[(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) \neq \emptyset\]

or

\[L_1 \cap \overline{L_2} \neq \emptyset\]

or

\[\overline{L_1} \cap L_2 \neq \emptyset\]

\[L_1 \subsetneq L_2\]

\[L_2 \subsetneq L_1\]

\[L_1 \neq L_2\]
Therefore, we only need to determine whether

\[ L(M) = (L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) = \emptyset \]

which is a solvable problem for DFAs:

\[ \text{EMPTY}_{\text{DFA}} \]
Undecidable Languages

Undecidable language = not decidable language

There is no decider:

there is no Turing Machine which accepts the language and makes a decision (halts) for every input string

(machine may make decision for some input strings)
For an undecidable language, the corresponding problem is undecidable (unsolvable):

there is no Turing Machine (Algorithm) that gives an answer (yes or no) for every input instance

(answer may be given for some input instances)
We have shown before that there are undecidable languages:

$L$ is Turing-Acceptable and undecidable.
We will prove that two particular problems are unsolvable:

- Membership problem
- Halting problem
Membership Problem

Input:
• Turing Machine $M$
• String $w$

Question: Does $M$ accept $w$?

$w \in L(M)$?

Corresponding language:

$A_{TM} = \{ \langle M, w \rangle : M$ is a Turing machine that accepts string $w \}$
Theorem: \( \mathcal{A}_{TM} \) is undecidable

(The membership problem problem is unsolvable)

Proof:

Basic idea:

We will assume that \( \mathcal{A}_{TM} \) is decidable;

We will then prove that every decidable language is Turing-Acceptable

\( A \) contradiction!
Suppose that $A_{TM}$ is decidable.
Let $L$ be a Turing recognizable language

Let $M_L$ be the Turing Machine that accepts $L$

We will prove that $L$ is also decidable:

we will build a decider for $L$
String description of $M_L$

Decider for $L$

Decider for $A_{TM}$

$M_L$ accepts $s$?

Input string $s$

Accept $s$ (and halt)

Reject $s$ (and halt)
Therefore, $L$ is decidable.

Since $L$ is chosen arbitrarily, every Turing-Acceptable language is decidable.

But there is a Turing-Acceptable language which is undecidable.

Contradiction!!!!

END OF PROOF
We have shown:

Undecidable $A_{TM}$

Decidable
We can actually show:

\[ A_{TM} \]

Decidable

Turing-Acceptable
\( A_{TM} \) is Turing-Acceptable

Turing machine that accepts \( A_{TM} \):

1. Run \( M \) on input \( w \)
2. If \( M \) accepts \( w \) then accept \( \langle M, w \rangle \)
Halting Problem

Input:
• Turing Machine $M$
• String $w$

Question: Does $M$ halt while processing input string $w$?

Corresponding language:

$HALT_{TM} = \{ \langle M, w \rangle : M \text{ is a Turing machine that halts on input string } w \}$
Theorem:  $\text{HALT}_{TM}$ is undecidable

(The halting problem is unsolvable)

Proof:

Basic idea:
Suppose that $\text{HALT}_{TM}$ is decidable; we will prove that every decidable language is also Turing-Acceptable
A contradiction!
Suppose that $\text{HALT}_{TM}$ is decidable.

Input string $\langle M, w \rangle$.

- If $M$ halts on input $w$, the decision is YES.
- If $M$ doesn't halt on input $w$, the decision is NO.
Let $L$ be a Turing-Acceptable language

Let $M_L$ be the Turing Machine that accepts $L$

We will prove that $L$ is also decidable:

we will build a decider for $L$
Decider for \( L \)

Input string \( S \)

Decider for \( \text{HALT}_{TM} \)

\( M_L \) halts on \( S \) ?

YES

Run \( M_L \) with input \( S \)

\( M_L \) halts and accepts

accept \( S \) and halt

\( M_L \) halts and rejects

reject \( S \) and halt

\( M_L \) halts on \( S \)

reject \( S \) and halt
Therefore, $L$ is decidable.

Since $L$ is chosen arbitrarily, every Turing-Acceptable language is decidable.

But there is a Turing-Acceptable language which is undecidable.

Contradiction!!!!

END OF PROOF
An alternative proof

Theorem: $HALT_{TM}$ is undecidable

(The halting problem is unsolvable)

Proof:

Basic idea:
Assume for contradiction that the halting problem is decidable;
we will obtain a contradiction using a diagonalization technique
Suppose that $\text{HALT}_{TM}$ is decidable.

Input string $\langle M, w \rangle$

Decider for $\text{HALT}_{TM}$

$H$

$\langle M \rangle \rightarrow \text{YES}$  $M$ halts on $w$

$w \rightarrow \text{NO}$  $M$ doesn't halt on $w$
Looking inside $H$

Decider for $HALT_{TM}$

Input string: $\langle M, w \rangle$

$H$

$q_0$ $M$ halts on $w$?

$q_{accept}$ YES

$q_{reject}$ NO
Construct machine $H'$:

If $M$ halts on input $w$ Then Loop Forever
Else Halt
Construct machine $F$:

If $M$ halts on input $\langle M \rangle$

Then loop forever

Else halt
Run $F$ with input itself

If $F$ halts on input $\langle F \rangle$

Then $F$ loops forever on input $\langle F \rangle$

Else $F$ halts on input $\langle F \rangle$

CONTRADICTION!!!

END OF PROOF
We have shown:

Undecidable $\text{HALT}_{\text{TM}}$

Decidable
We can actually show:

- Turing-Acceptable
- $\text{HALT}_{TM}$
- Decidable
$\text{HALT}_{TM}$ is Turing-Acceptable

Turing machine that accepts $\text{HALT}_{TM}$:

1. Run $M$ on input $w$
2. If $M$ halts on $w$ then accept $\langle M, w \rangle$