Undecidability of the halting problem

$$A_{TM} = \{\langle M, w\rangle \mid M \textit{ is a Turing machine that accepts } w\}$$

Theorem

A_{TM} is undecidable

Undecidability of the halting problem

$$A_{TM} = \{\langle M, w\rangle \mid M \textit{ is a Turing machine that accepts } w\}$$

Theorem

A_{TM} is undecidable

Proof.

1. Assume that A_{TM} is decidable by a Turing machine H
2. Construct a new Turing machine D which takes $\langle M\rangle$ as input and works as follows:
    - Run H on $\langle M, \langle M\rangle\rangle$ and output the opposite of what H outputs
3. Running D on input $\langle D\rangle$ results in a contradiction because D rejects $\langle D\rangle$ if D accepts $\langle D\rangle$, and D accepts $\langle D\rangle$ if D rejects $\langle D\rangle$

$\square$

An explicit language which is not Turing-recognizable

$$\overline{A_{TM}} = \{w \mid w \notin A_{TM}\}$$

An explicit language which is not Turing-recognizable

$$\overline{A_{TM}} = \{w \mid w \notin A_{TM}\}$$

Theorem

$\overline{A_{TM}}$ is not Turing-recognizable

An explicit language which is not Turing-recognizable

$$\overline{A_{TM}} = \{w \mid w \notin A_{TM}\}$$

Theorem

$\overline{A_{TM}}$ is not Turing-recognizable

Proof.

1. Assume with the aim of reaching a contradiction that $\overline{A_{TM}}$ is Turing-recognizable.

An explicit language which is not Turing-recognizable

$$\overline{A_{TM}} = \{w \mid w \notin A_{TM}\}$$

Theorem

$\overline{A_{TM}}$ is not Turing-recognizable

Proof.

1. Assume with the aim of reaching a contradiction that $\overline{A_{TM}}$ is Turing-recognizable.
2. Let M_1 be a Turing machine recognizing $\overline{A_{TM}}$ and M_2 a Turing machine recognizing A_{TM}.

An explicit language which is not Turing-recognizable

$$\overline{A_{TM}} = \{w \mid w \notin A_{TM}\}$$

Theorem

$\overline{A_{TM}}$ is not Turing-recognizable

Proof.

1. Assume with the aim of reaching a contradiction that $\overline{A_{TM}}$ is Turing-recognizable.
2. Let M_1 be a Turing machine recognizing $\overline{A_{TM}}$ and M_2 a Turing machine recognizing A_{TM}.
3. On input w run both M_1 and M_2 on w in parallel

An explicit language which is not Turing-recognizable

$$\overline{A_{TM}} = \{w \mid w \notin A_{TM}\}$$

Theorem

$\overline{A_{TM}}$ is not Turing-recognizable

Proof.

1. Assume with the aim of reaching a contradiction that $\overline{A_{TM}}$ is Turing-recognizable.
2. Let M_1 be a Turing machine recognizing $\overline{A_{TM}}$ and M_2 a Turing machine recognizing A_{TM}.
3. On input w run both M_1 and M_2 on w in parallel
4. If M_1 accepts, then reject. If M_2 accepts, then accept.
5. This shows that A_{TM} is decidable, which is a contradiction

□

Reductions

Reductions

Definition

A function $f : \Sigma^* \to \Sigma^*$ is computable if some Turing machine M on every input w halts with $f(w)$ on its tape

Reductions

Definition

Language A is mapping reducible to language B if there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$ such that for every w

$$w \in A \text{ iff } f(w) \in B$$

The function f is called a reduction from A to B
If A is mapping reducible to B then we write $A \leq_m B$

Reductions

Theorem

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

Reductions

Theorem

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

Proof.

Let M_B be a Turing machine that decides B and f the reduction from A to B. Given input w:

1. Compute $f(w)$
2. Run M_B on $f(w)$, accept if M_B accepts $f(w)$, and reject if M_B rejects $f(w)$

□

Reductions

Theorem

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

Reductions

Theorem

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

Proof.

Assume with the aim of reaching a contradiction that B is decidable. Let M_B be a Turing machine that decides B and f the reduction from A to B. Given input w:

1. Compute $f(w)$
2. Run M_B on $f(w)$, accept if M_B accepts $f(w)$, and reject if M_B rejects $f(w)$
3. This shows that A is decidable, which is a contradiction

□

Reductions

Theorem

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

Reductions

Theorem

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

Reductions

$$E_{TM} = \{\langle M \rangle \mid M \textit{ is a Turing machine and } L(M) = \emptyset\}$$

Reductions

$$E_{TM} = \{\langle M \rangle \mid M \textit{ is a Turing machine and } L(M) = \emptyset\}$$

Theorem

E_{TM} is undecidable

Reductions

$$E_{TM} = \{\langle M \rangle \mid M \textit{ is a Turing machine and } L(M) = \emptyset\}$$

Theorem

E_{TM} is undecidable

Proof.

By reduction from A_{TM} (Theorem 5.2 in Sipser) □

Reductions

$$EQ_{TM} = \{\langle M_1, M_2\rangle \mid M_1 \textit{ and } M_2 \textit{ are TMs and } L(M_1) = L(M_2)\}$$

Reductions

$$EQ_{TM} = \{\langle M_1, M_2\rangle \mid M_1 \textit{ and } M_2 \textit{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem

EQ_{TM} is undecidable

Reductions

$$EQ_{TM} = \{\langle M_1, M_2\rangle \mid M_1 \textit{ and } M_2 \textit{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem

EQ_{TM} is undecidable

Proof.

$E_{TM} \leq_m EQ_{TM}$:

Reductions

$$EQ_{TM} = \{\langle M_1, M_2\rangle \mid M_1 \textit{ and } M_2 \textit{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem

EQ_{TM} *is undecidable*

Proof.

$E_{TM} \leq_m EQ_{TM}$:

1. Let M_2 be a Turing machine such that $L(M_2) = \emptyset$

Reductions

$$EQ_{TM} = \{\langle M_1, M_2\rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem

EQ_{TM} is undecidable

Proof.

$E_{TM} \leq_m EQ_{TM}$:

1. Let M_2 be a Turing machine such that $L(M_2) = \emptyset$
2. Given a Turing machine M, let $f(\langle M\rangle) = \langle M, M_2\rangle$

Reductions

$$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$$

Theorem

EQ_{TM} is undecidable

Proof.

$E_{TM} \leq_m EQ_{TM}$:

1. Let M_2 be a Turing machine such that $L(M_2) = \emptyset$
2. Given a Turing machine M, let $f(\langle M \rangle) = \langle M, M_2 \rangle$
3. $\langle M \rangle \in E_{TM}$ iff $L(M) = \emptyset$ iff $L(M) = L(M_2)$ iff $\langle M, M_2 \rangle \in EQ_{TM}$

□

Incompleteness Theorem via Undecidability

Kurt Gödel (1906-1978)

Incompleteness Theorem via Undecidability (S, 6.2)

Theorem

(Informally) There are true mathematical statements that cannot be proved

Incompleteness Theorem via Undecidability (S, 6.2)

Theorem

(Informally) There are true mathematical statements that cannot be proved

Proof idea.

1. The language of all true mathematical statements is undecidable (by reduction from A_{TM})

Incompleteness Theorem via Undecidability (S, 6.2)

Theorem

(Informally) There are true mathematical statements that cannot be proved

Proof idea.

1. The language of all true mathematical statements is undecidable (by reduction from A_{TM})
2. The language of all provable statements is Turing recognizable by a Turing machine M

Incompleteness Theorem via Undecidability (S, 6.2)

Theorem

(Informally) There are true mathematical statements that cannot be proved

Proof idea.

1. The language of all true mathematical statements is undecidable (by reduction from A_{TM})
2. The language of all provable statements is Turing recognizable by a Turing machine M
3. Assume that all true statements are provable

Incompleteness Theorem via Undecidability (S, 6.2)

Theorem

(Informally) There are true mathematical statements that cannot be proved

Proof idea.

1. The language of all true mathematical statements is undecidable (by reduction from A_{TM})
2. The language of all provable statements is Turing recognizable by a Turing machine M
3. Assume that all true statements are provable
4. Given a statement Φ, run M in parallel on Φ and $\neg\Phi$. One of them is true and thus (by assumption) provable. If Φ is provable then Φ is true and if $\neg\Phi$ is provable then Φ is false.

Incompleteness Theorem via Undecidability (S, 6.2)

Theorem

(Informally) There are true mathematical statements that cannot be proved

Proof idea.

1. The language of all true mathematical statements is undecidable (by reduction from A_{TM})
2. The language of all provable statements is Turing recognizable by a Turing machine M
3. Assume that all true statements are provable
4. Given a statement Φ, run M in parallel on Φ and $\neg\Phi$. One of them is true and thus (by assumption) provable. If Φ is provable then Φ is true and if $\neg\Phi$ is provable then Φ is false.
5. So M decides the truth of Φ. This is a contradiction (with 1 above)

□