# Halting Problem is undecidable

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

One attempt :

$U$ = On input $\langle M, w\rangle$. $M$ is a TM and $w$ is a string.

1. Simulate $M$ on $w$.
2. If $M$ accepts, accept.
   If $M$ rejects, reject.

Not happening

This is only a recognizer as it will loop, when $M$ loops on $w$. A decider must reject $\langle M, w\rangle$ when $M$ loops on $w$ as well.

$A_{TM}$ is Turing recognizable. However, if the TM $U$ knows that $M$ is going to get stuck in a loop with $w$, $U$ can reject $w$. This is the hardest part.

Real Halting Problem

$HALT_{TM} = \{\langle M, w\rangle \mid M$ is a TM and $M$ halts on $w\}$

$HALT_{TM}$ is also undecidable. We will also refer to $A_{TM}$ as the "halting problem".

---

Suppose $A_{TM}$ is decidable. Then there is a TM H that takes $\langle M, w \rangle$ as input and accepts if M accepts w and rejects if M does not accept w. H must always halt.

(1) M rejects w
(2) M loops on w

Theorem 4.11: $A_{TM}$ is undecidable.

Proof: Consider the TM D: On input $\langle M \rangle$ where $\langle M \rangle$ is the description of a TM.

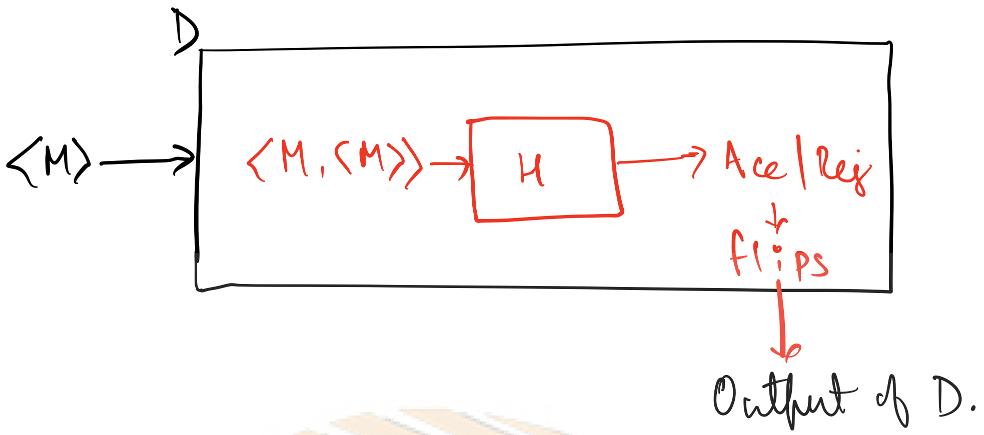
D does this: 1. Run H on $\langle M, \langle M \rangle \rangle$.

takes $\langle M \rangle$ as input

View $\langle M \rangle$ as an input string.

2. If H accepts, reject
If H rejects, accept.

Flips the o/p.




Let $D(\langle M \rangle)$ denote what $D$ does on input $\langle M \rangle$.

$$D(\langle M \rangle) = \begin{cases} \text{Accept if } M \text{ does not accept } \langle M \rangle \\ \text{Reject if } M \text{ accepts } \langle M \rangle. \end{cases}$$

reject
loop.

What does D do when given ⟨D⟩ as input?

$$D(\langle D \rangle) = \begin{cases} \text{Accept if } D \text{ does not accept } \langle D \rangle \\ \text{Reject if } D \text{ accepts } \langle D \rangle. \end{cases}$$

So D does on ⟨D⟩, the opposite of what D is supposed to do on ⟨D⟩. This is a contradiction.

---

<u>Why is this diagonalization?</u>

Let $A_{TM}$ be decidable. Let us build the following table. As noted before, we can enumerate all the Turing machines $M_1, M_2, M_3 \dots$

| Machine \ Input | $\langle M_1 \rangle$ | $\langle M_2 \rangle$ | $\langle M_3 \rangle$ | $\langle M_4 \rangle \dots$ |
|---|---|---|---|---|
| $M_1$ | Accept | Reject | Accept | Loop |
| $M_2$ | Accept | Accept | Reject | Loop |
| $M_3$ | Reject | Accept | Accept | Accept |
| $M_4$ | Loop | Accept | Loop | Reject |
| $\vdots$ | | | | |

What does $H$ do $\langle M_i, \langle M_j \rangle \rangle$ is given as input?

Same table as above, but all the "Loop" will be replaced by "Reject". When $M_i$ does not accept $\langle M_j \rangle$, then $H$ will reject $\langle M_i, \langle M_j \rangle \rangle$.

$A_{TM}$ is decidable $\Rightarrow$ $H$ exists

Let us draw the table for $H$ on $\langle M_i, \langle M_j \rangle \rangle$.

3.012
2.362
5.243...
0.174...

| $\langle M_i, \langle M_j \rangle \rangle$ | $\langle M_1 \rangle$ | $\langle M_2 \rangle$ | $\langle M_3 \rangle$ | $\langle M_4 \rangle$ | ... | $\langle D \rangle$ |
|---|---|---|---|---|---|---|
| $M_1$ | Acc | Rej | Acc | Rej | ... | |
| $M_2$ | Acc | Acc | Rej | Rej | ... | |
| $M_3$ | Rej | Acc | Acc | Acc | ... | |
| $M_4$ | Rej | Acc | Rej | Rej | ... | |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | | |
| $D$ | Rej | Rej | Rej | Acc | ... | ? |

What's this entry?

$A_{TM}$ is decidable $\Rightarrow$ $H$ is a TM

$\Rightarrow$ $D$ is a TM

Since TM's can be enumerated, $D$ is in the list as some $M_i$. But $D$ on $\langle M \rangle$, flips the action of $H$ on $\langle M \rangle$.

What does $D$ do on $\langle D \rangle$? Contradiction!

Def: $A$ is co-Turing recognizable if $\bar{A}$ is Turing recognizable.

That is, $\exists$ a TM $M$ such that $L(M) = \bar{A}$. We can flip the output of $M$ to get a TM $M'$, such that $M'$ always halts and rejects when $w \notin A$. (But $L(M')$ need not be $A$)

| | | |
|---|---|---|
| $\bar{A}$ | M accepts | Rej |
| $A$ | M reject | Acc |
| $A$ | M loops | Loops |

When $x \in \bar{A} \iff x \notin A$, $M$ accepts $x$

$x \notin \bar{A} \iff x \in A$, $M$ can reject $x$

OR $M$ can loop on $x$.

Theorem 4.22: A language is decidable $\iff$ it is Turing recognizable and co-Turing recognizable.

Proof: $A$ is decidable $\Rightarrow$ $\exists$ a TM $M$, decider with $L(M) = A$

$\Rightarrow$ $A$ is Turing recognizable.

$M$ always halts. So flip the output of $M$ to get $M'$, a decider for $\bar{A}$.

$\Rightarrow$ $\bar{A}$ is Turing recognizable.

$\Rightarrow$ $A$ is co-Turing recognizable.

Suppose $A$ is both Turing recognizable and co-Turing recognizable.

$\exists$ TM $M$ such that $L(M) = A$

$\exists$ TM $M'$ such that $L(M') = \bar{A}$

Now we will see how to build a decider for $A$. Given input $w$, run both $M$ and $M'$ in parallel (one step each). Given any $w$, it is either in $A$ or $\bar{A}$. If it is in $A$, $M$ accepts $w$. If it is in $\bar{A}$, $M'$ accepts $w$. So for any $w$, either $M$ or $M'$ must halt.

$\boxed{M}$

If $M$ accepts, accept.

If $M'$ accepts, reject.

$\boxed{M'}$

---

Cor 4.23: $\overline{A_{TM}}$ is not Turing recognizable.

Proof: $A_{TM}$ is undecidable.

$A_{TM}$ is Turing recognizable.

If $\overline{A_{TM}}$ was Turing recognizable, then by Theorem 4.22 above, $A_{TM}$ will be decidable. Hence $\overline{A_{TM}}$

is not Turing recognizable. (ATM is not co-Turing recognizable)

---

→ A, B, is L(A) = L(B)?

$EQ_{CFG}$ is co-Turing recognizable.

Given CFG's A and B, we can run $A_{CFG}$ for each $w \in \Sigma^*$ till you find a $w$ that is generated by one and not the other.

Fact (without proof): $EQ_{CFG}$ is undecidable.

Assuming this, it follows that $EQ_{CFG}$ is not Turing recognizable as well.

If $EQ_{CFG}$ was Turing recognizable, then Theorem 4.22 implies that $EQ_{CFG}$ is decidable, which we know is not the case.

Hence $EQ_{CFG}$ is not Turing recognizable.