CMSC 28100

Introduction to Complexity Theory

Autumn 2025

Instructor: William Hoza

Which problems

can be solved

through computation?

What are Turing machines capable of?

Which languages are decidable?

The halting problem

- **Informal problem statement:** Given a Turing machine M and an input w, determine whether M halts on w.
- **The same problem, formulated as a language:**

$$\mathrm{HALT} = \{\langle M, w\rangle : M \text{ is a Turing machine that halts on input } w\}$$

- It's the problem of identifying bugs in someone else's code! 🐞

Attempting to decide HALT

- Given $\langle M, w \rangle$:
  1. Simulate M on w
  2. If it halts, accept
  3. Otherwise, reject

Does the proposed algorithm work?

A: No. It's not necessarily possible to simulate M on w

B: No. There are inputs for which it should accept, but it doesn't

C: No. There are inputs for which it should reject, but it doesn't

D: Yes. HALT is decidable

Respond at PollEv.com/whoza or text "whoza" to 22333

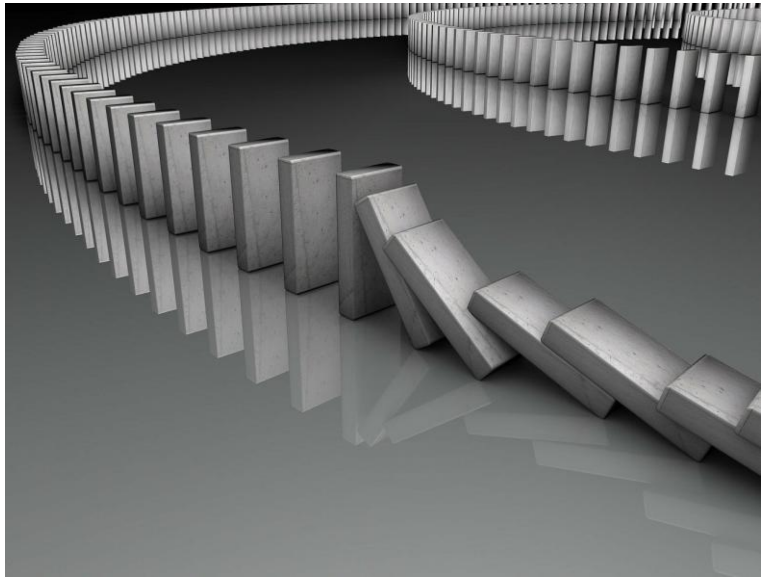
The halting problem is undecidable

- $\text{HALT} = \{\langle M, w\rangle : M \text{ is a Turing machine that halts on } w\}$

Theorem: HALT is undecidable

- How should we prove it?

Reductions



- We already proved that SELF-REJECTORS is undecidable
- Plan: Let's show that if HALT were decidable, then SELF-REJECTORS would also be decidable – a contradiction
- "Reduction from SELF-REJECTORS to HALT"

Proof that HALT is undecidable

- Assume for the sake of contradiction that there is some Turing machine H that decides HALT
- Let's construct a new TM S that decides $\mathrm{SELF\text{-}REJECTORS}$

S

> Given the input $\langle M \rangle$:
>
> 1. Simulate H on $\langle M, \langle M \rangle \rangle$
> 2. If H rejects, reject. Otherwise:
> 3. Simulate M on $\langle M \rangle$
> 4. If M rejects, accept; if M accepts, reject.

- If M loops on $\langle M \rangle$, then H rejects, so S rejects
- If M accepts $\langle M \rangle$, then H accepts and M accepts, so S rejects
- If M rejects $\langle M \rangle$, then H accepts and M rejects, so S accepts

Reductions

- Our goal was to prove that HALT is undecidable
- Our strategy was to design an algorithm for deciding SELF-REJECTORS!

  (using a hypothetical device that decides HALT)
- The existence of one algorithm implies the non-existence of another!

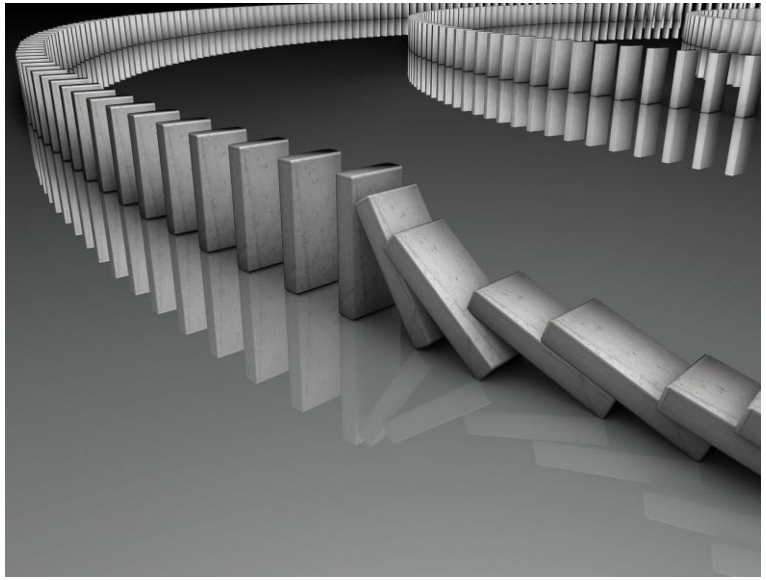
Note on standards of rigor

Given the input $\langle M \rangle$:

1. Simulate H on $\langle M, \langle M \rangle \rangle$
2. If H rejects, reject. Otherwise:
3. Simulate M on $\langle M \rangle$
4. If M rejects, accept; if M accepts, reject.

- Going forward, when we want to construct a Turing machine (e.g., for a reduction), we will simply describe what it does in plain English
  - As if we were giving instructions to a human being
  - Plain English description can be formalized as a Turing machine, but this is tedious
  - You should follow this convention on Exercise 8 and beyond

Undecidability



- Now we have two examples of undecidable languages
- SELF-REJECTORS and HALT
- Next, we will see an example of an undecidable language that (seemingly) isn't about Turing machines

Post's Correspondence Problem

- **Given:** A list of "dominos" $\begin{bmatrix} t_1 \\ b_1 \end{bmatrix}, \begin{bmatrix} t_2 \\ b_2 \end{bmatrix}, \begin{bmatrix} t_3 \\ b_3 \end{bmatrix}, \cdots, \begin{bmatrix} t_k \\ b_k \end{bmatrix}$, where $t_i, b_i \in \Gamma^*$ for some alphabet Γ

- **Goal:** Determine whether it is possible to construct a "match"

- A "match" is a sequence of dominos $\begin{bmatrix} t_{i_1} \\ b_{i_1} \end{bmatrix}\begin{bmatrix} t_{i_2} \\ b_{i_2} \end{bmatrix}\begin{bmatrix} t_{i_3} \\ b_{i_3} \end{bmatrix}\begin{bmatrix} t_{i_4} \\ b_{i_4} \end{bmatrix}\begin{bmatrix} t_{i_5} \\ b_{i_5} \end{bmatrix} \cdots \begin{bmatrix} t_{i_n} \\ b_{i_n} \end{bmatrix}$

such that $t_{i_1} t_{i_2} \cdots t_{i_n} = b_{i_1} b_{i_2} \cdots b_{i_n}$

- Using the same domino multiple times is permitted

Post’s Correspondence Problem: Example 1

- Suppose we are given

0	1	111	0
1	0	1	000

- This is a YES case. Match:

111	0	0	0	← 111000
1	1	1	000	← 111000

Post’s Correspondence Problem: Example 2

- Suppose we are given

$+5 =$	$6 = 7$	$3 +$	$1 +$	$4 =$	$4 = 2$
$= 1 +$	7	$3 + 4 =$	$6 =$	$1 + 6$	$2 + 5$

- This is a YES case. Match:

$3 +$	$4 = 2$	$+5 =$	$1 +$	$6 = 7$	$\leftarrow 3 + 4 = 2 + 5 = 1 + 6 = 7$
$3 + 4 =$	$2 + 5$	$= 1 +$	$6 =$	7	$\leftarrow 3 + 4 = 2 + 5 = 1 + 6 = 7$

Post’s Correspondence Problem: Example 3

- Suppose we are given

  $\left[\dfrac{\#}{\#\$}\right] \quad \left[\dfrac{\#\$\#}{\$\#}\right] \quad \left[\dfrac{\$}{\#\$}\right]$

- This is a NO case
- **Proof:** A match would have to start with $\left[\dfrac{\#}{\#\$}\right]$ …
- …which means there will always be more \$ symbols on the bottom than on the top

Post’s Correspondence Problem is undecidable

- **Post’s correspondence problem, formulated as a language:**

$$\mathrm{PCP} = \{\langle t_1, \dots, t_k, b_1, \dots, b_k \rangle : \exists i_1, \dots, i_n \text{ such that } t_{i_1} \cdots t_{i_n} = b_{i_1} \cdots b_{i_n}\}$$

Theorem: PCP is undecidable

- Proof on the upcoming 18 slides. Outline:
    - Step 1: Reduce HALT to a modified version (“MPCP”)
    - Step 2: Reduce MPCP to PCP

Modified PCP

$$\mathrm{MPCP} = \{\langle t_1, \ldots, t_k, b_1, \ldots, b_k\rangle : \exists i_1, \ldots, i_n \text{ such that } t_1 t_{i_1} \cdots t_{i_n} = b_1 b_{i_1} \cdots b_{i_n}\}$$

- New feature: In MPCP, matches must start with the first domino
- We'll use a double outline to indicate the special first domino: $\boxed{\boxed{\begin{matrix} t_1 \\ b_1 \end{matrix}}}$

Lemma: MPCP is undecidable

Proof that MPCP is undecidable

- Assume there is a TM P that decides MPCP
- Let's construct a new TM H that decides HALT

H

Given $\langle M, w\rangle$:

1. Construct dominos $t_1, \dots, t_k, b_1, \dots b_k$ based on M and w

   (details on upcoming slides)

2. Simulate P on $\langle t_1, \dots, t_k, b_1, \dots, b_k\rangle$
3. If P accepts, accept. If P rejects, reject.

Reducing HALT to MPCP

- We are given $\langle M, w \rangle$, where

$$M = \left(Q, q_0, q_{\text{accept}}, q_{\text{reject}}, \Sigma, \sqcup, \delta\right)$$

- Our job is to produce a collection of dominos
- Plan: Produce dominos such that constructing a match is equivalent to constructing a halting computation history

The dominos for

Given $\langle M, w \rangle$, how does one construct these dominos?

A: Simulate M on w. If it accepts, accept; if it rejects, reject

B: Simulate M on w and copy whatever dominos it produces

C: There is no algorithm for constructing the dominos

D: Inspect the transition function of M

Respond at PollEv.com/whoza or text "whoza" to 22333

- $\boxed{\boxed{\begin{matrix}\epsilon \\ (q_0 \sqcup w)\end{matrix}}}$, $\boxed{\begin{matrix}(\\ (\end{matrix}}$, $\boxed{\begin{matrix}) \\ )\end{matrix}}$, $\boxed{\begin{matrix}(q_{\text{accep}} \\ \epsilon\end{matrix}}$
- For every $q \in Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\}$ and every $b \in \Sigma$:
  - If $\delta(q, b) = (q', b', \mathrm{R})$, we include $\boxed{\begin{matrix}qb) \\ b'q' \sqcup)\end{matrix}}$, and we include $\boxed{\begin{matrix}qba \\ b'q'a\end{matrix}}$ for every $a \in \Sigma$
  - If $\delta(q, b) = (q', b', \mathrm{L})$, we include $\boxed{\begin{matrix}(qb \\ (q' \sqcup b'\end{matrix}}$, and we include $\boxed{\begin{matrix}aqb \\ q'ab'\end{matrix}}$ for every $a \in \Sigma$
- $\boxed{\begin{matrix}b \\ b\end{matrix}}$, $\boxed{\begin{matrix}bq_{\text{accept}} \\ q_{\text{accept}}\end{matrix}}$, $\boxed{\begin{matrix}q_{\text{accept}}b \\ q_{\text{accept}}\end{matrix}}$, $\boxed{\begin{matrix}q_{\text{reject}}b \\ q_{\text{reject}}\end{matrix}}$, and $\boxed{\begin{matrix}bq_{\text{reject}} \\ q_{\text{reject}}\end{matrix}}$ for every $b \in \Sigma$

Proof that MPCP is undecidable

- Assume there is a TM P that decides MPCP
- Let's construct a new TM H that decides HALT

H

Given $\langle M, w \rangle$:

1. Construct dominos $t_1, \dots, t_k, b_1, \dots b_k$ based on M and w (details on preceding slides)
2. Simulate P on $\langle t_1, \dots, t_k, b_1, \dots, b_k \rangle$
3. If P accepts, accept. If P rejects, reject.

Need to show:

- If M halts on w, then there is a match
- If there is a match, then M halts on w

Domino Feature 1

- **Domino Feature 1:** For every non-halting configuration C of M, there is a sequence of dominos such that the top string is (C) and bottom string is $\big(\mathrm{NEXT}(C)\big)$
- Proof omitted, but here's an example:

(	0	1	#	0	$0q_1 0$	0	$\sqcup$	)
(	0	1	#	0	$q_2 01$	1	$\sqcup$	)

- Think of this sequence as one "super-domino"

$$(C)$$
$$\big(\mathrm{NEXT}(C)\big)$$

If M halts on w, then there is a match

- Let $C_0, \ldots, C_T$ be the halting computation history of M on w
- Partial match:

ϵ	(C_0)	(C_1)	$\cdots$	(C_{T-1})
(C_0)	(C_1)	(C_2)		(C_T)

- At this point, we have an extra (C_T) on the bottom

Domino Feature 2

- **Domino Feature 2:** For every halting configuration D, there is a sequence of dominos such that the top string is (D) and the bottom string is (D'), where D' is a halting configuration* of length $|D| - 1$
  - *Possibly $D' = q_{\text{accept}}$ or q_{reject} by itself
- Proof omitted, but here's an example:

(	0	1	#	0	$0q_{\text{reject}}$	0	$\sqcup$	)
(	0	1	#	0	q_{reject}	1	$\sqcup$	)

- Think of this sequence as one "super domino"

(D)
(D')

If M halts on w, then there is a match

- We construct a sequence of shorter and shorter halting configurations $C_T = D_0, D_1, \dots, D_n$ such that $|D_n| = 1$ and we have a super-domino $\boxed{\begin{matrix}(D_{i-1})\\(D_i)\end{matrix}}$ for every i
- Full match:

ϵ	(C_0)	(C_1)	…	(C_{T-1})	(C_T)	(D_1)	(D_2)	…	(D_{n-1})	(D_n)
(C_0)	(C_1)	(C_2)		(C_T)	(D_1)	(D_2)	(D_3)		(D_n)	ϵ

Proof that MPCP is undecidable

- Assume there is a TM P that decides MPCP
- Let's construct a new TM H that decides HALT

H

Given $\langle M, w\rangle$:

1. Construct dominos $t_1, \ldots, t_k, b_1, \ldots b_k$ based on M and w (details on preceding slides)
2. Simulate P on $\langle t_1, \ldots, t_k, b_1, \ldots, b_k\rangle$
3. If P accepts, accept. If P rejects, reject.

Need to show:

- If M halts on w, then there is a match ✔
- If there is a match, then M halts on w