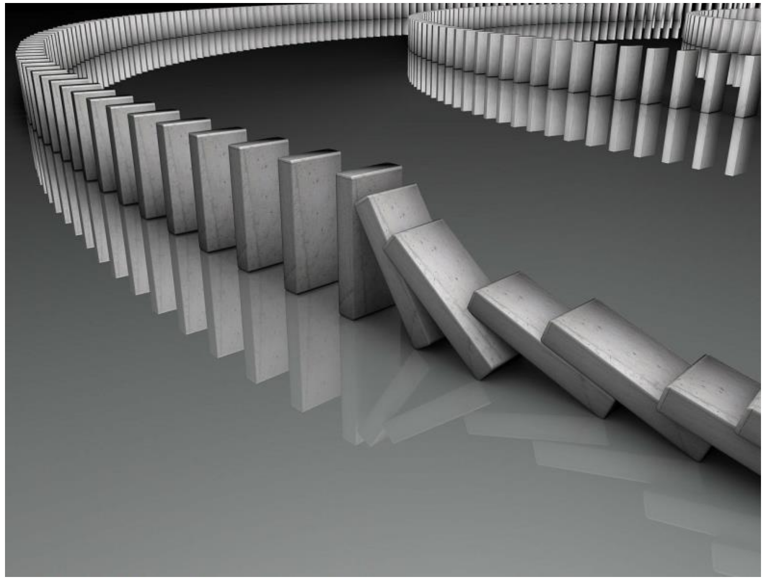
CMSC 28100

Introduction to Complexity Theory

Spring 2024
Instructor: William Hoza

Which languages are decidable?

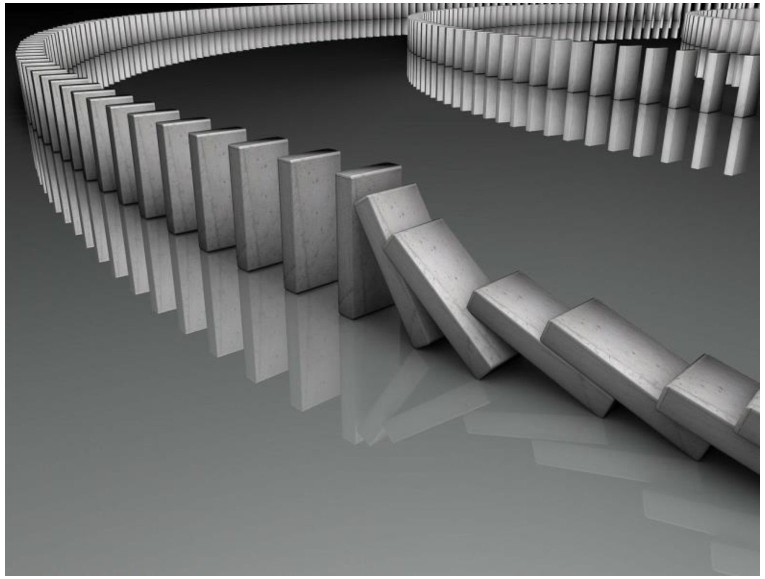
Undecidability



- We have seen several examples of undecidable languages
- SELF-REJECTORS, HALT, $\overline{\text{HALT}}$, A_{TM}, E_{TM}

Undecidability beyond analysis of TMs

- So far, every undecidable problem we have seen has been a problem about analyzing the behavior of a given Turing machine
  - Does it halt on such-and-such input?
  - Is there any input it accepts?
  - Etc.
- What else is undecidable?



Post's Correspondence Problem

- **Given:** An alphabet Λ and two sequences of strings $b_1, \ldots, b_k, t_1, \ldots, t_k \in \Lambda^*$
- **Goal:** Determine whether there exists a sequence of indices $i_1, \ldots, i_n$ such that

$$t_{i_1} t_{i_2} \cdots t_{i_k} = b_{i_1} b_{i_2} \cdots b_{i_k}$$

Post's Correspondence Problem

- Helpful picture: We are given a set of "dominos"

t_1	t_2	t_3	…	t_k
b_1	b_2	b_3		b_k

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

t_{i_1}	t_{i_2}	t_{i_3}	t_{i_4}	t_{i_5}	…	t_{i_n}
b_{i_1}	b_{i_2}	b_{i_3}	b_{i_4}	b_{i_5}		b_{i_n}

in which the sequence of symbols on top equals the sequence of symbols on the bottom

- Using the same domino multiple times is permitted

Post's Correspondence Problem: Example 1

- Suppose we are given

$$\left[\begin{array}{c}0\\1\end{array}\right] \quad \left[\begin{array}{c}1\\0\end{array}\right] \quad \left[\begin{array}{c}11\\\epsilon\end{array}\right] \quad \left[\begin{array}{c}\epsilon\\00\end{array}\right]$$

- This is a YES case. Match:

$$\left[\begin{array}{c}11\\\epsilon\end{array}\right]\left[\begin{array}{c}0\\1\end{array}\right]\left[\begin{array}{c}0\\1\end{array}\right]\left[\begin{array}{c}\epsilon\\00\end{array}\right] \quad \begin{array}{l}\leftarrow 1100\\ \leftarrow 1100\end{array}$$

Post’s Correspondence Problem: Example 2

- Suppose we are given

MP	IO	N	CO	UT	AT
OM	T	ION	C	PU	TA

- This is a YES case because there is a match:

CO	MP	UT	AT	IO	N
C	OM	PU	TA	T	ION

← COMPUTATION
← COMPUTATION

- …and another match:

CO	MP	UT	IO	N
C	OM	PU	T	ION

← COMPUTION
← COMPUTION

Post’s Correspondence Problem: Example 3

- Suppose we are given

$$\boxed{\begin{matrix}0\\01\end{matrix}} \quad \boxed{\begin{matrix}010\\10\end{matrix}} \quad \boxed{\begin{matrix}1\\01\end{matrix}}$$

- This is a NO case. Proof: A match would have to start with $\boxed{\begin{matrix}0\\01\end{matrix}}$,

and consequently, we will always have more ones on the bottom than on the top

Post's Correspondence Problem is undecidable

- Define

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

Theorem: PCP is undecidable

- Proof outline:
  - Step 1: Show that a modified version ("MPCP") is undecidable by reduction from HALT
  - Step 2: Show that PCP is undecidable by reduction from MPCP

Modified PCP

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

- The difference between PCP and MPCP: In MPCP, matches must start with the first domino

Reduction from HALT to MPCP

- To show that MPCP is undecidable, we will design a mapping reduction $f(\langle M, w\rangle) = \langle \Lambda, t_1, \dots, t_k, b_1, \dots, b_k\rangle$
- We will ensure that:
  - If M halts on w, then there is a match ("YES maps to "YES")
  - If M loops on w, then there is no match ("NO maps to NO")
  - The function f is computable

Reduction from $\mathbf{HALT}$ to $\mathbf{MPCP}$

- For the reduction, we are given $\langle M, w \rangle$, where

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

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

Reduction from H

Given $\langle M, w \rangle$, how would we compute $f(\langle M, w \rangle)$?

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 does not exist an algorithm that computes f

D: Inspect the transition function of M to figure out the dominos

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

- We produce the following domi

- $\frac{\epsilon}{\Diamond q_0 w \sqcup \#}$, $\frac{\#}{\#}$, $\frac{\#}{\sqcup \#}$, $\frac{q_{ac}}{\epsilon}$, and $\frac{}{\epsilon}$

Reduction is computable ✔

- $\frac{qb}{b'q'}$ for every q, b, q', b' such that $\delta(q, b) = (q', b', \mathrm{R})$ and $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$

- $\frac{aqb}{q'ab'}$ for every q, b, q', b', a such that $\delta(q, b) = (q', b', \mathrm{L})$ and $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$

- $\frac{b}{b}$, $\frac{bq}{q}$, and $\frac{qb}{q}$ for every $b \in \Gamma$ and $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$

Domino feature 1

- Let $C = uqv$ be a configuration where $v \neq \epsilon$ and uv starts with $\Diamond$
- Fact: There is a sequence of dominos such that the top string is C and bottom string is $\mathrm{NEXT}(C)$
- Think of this sequence as one "super-domino" $\boxed{\begin{array}{c} C \\ \mathrm{NEXT}(C) \end{array}}$

YES maps to YES

- Let $C_0, \ldots, C_T$ be the halting computation history of M on w
- Let $C_i' = C_i \sqcup^i$, and note that $\mathrm{NEXT}(C_i' \sqcup) = C_{i+1}'$
- Partial match:

ϵ	$C_0' \sqcup$	$\#$	$C_1' \sqcup$	$\#$	…	$\#$	$C_{T-1}' \sqcup$	$\#$
$\Diamond q_0 w \sqcup \#$	C_1'	$\sqcup \#$	C_2'	$\sqcup \#$		$\sqcup \#$	C_T'	$\#$

- At this point, we have an extra $C_T'\#$ on the bottom

Domino feature 2

- Fact: For every halting configuration H with $|H| > 1$, there is a sequence of dominos such that the top string is H and the bottom string is a halting configuration H' with $|H'| = |H| - 1$ $\left[\dfrac{H}{H'}\right]$
- Proof: Use the $\left[\dfrac{b}{b}\right]$, $\left[\dfrac{bq}{q}\right]$, and $\left[\dfrac{qb}{q}\right]$ dominos to effectively "delete" one symbol from H

YES maps to YES

- Starting with $H_0 = C'_T$, we construct a sequence of shorter and shorter halting configurations $H_1, \dots, H_n$ such that we have a super-domino $\left[\frac{H_{i-1}}{H_i}\right]$ for every i, until eventually we reach $H_n \in \{q_{\text{accept}}, q_{\text{reject}}\}$

- Full match:

ϵ	$C'_0 \sqcup$	$\#$	$C'_1 \sqcup$	$\#$	$\dots$	$\#$	$C'_{T-1} \sqcup$	$\#$	H_0	$\#$	H_1	$\#$	$\dots$	$\#$	H_{n-1}	$\#$	$H_n\#$
$\Diamond q_0 w \sqcup \#$	C'_1	$\sqcup \#$	C'_2	$\sqcup \#$		$\sqcup \#$	C'_T	$\#$	H_1	$\#$	H_2	$\#$		$\#$	H_n	$\#$	ϵ

NO maps to NO

- Suppose M loops on w. Let $C_0, C_1, C_2, \ldots$ be the computation history of M on w (an infinite sequence of configurations)
- Assume, for the sake of contradiction, that there is a match