Theory of Formal Languages and Automata

Lecture 21

Mahdi Dolati

Sharif University of Technology

Fall 2023

December 15, 2023

A Simple Undecidable Problem

- Undecidability is not limited to problems concerning automata.
  - Post Correspondence Problem, or PCP.
- Definition:
  - A domino that has two sides: $\left[\frac{\mathtt{a}}{\mathtt{ab}}\right]$
  - A collection of dominos:

$$\left\{ \left[\frac{\mathtt{b}}{\mathtt{ca}}\right], \left[\frac{\mathtt{a}}{\mathtt{ab}}\right], \left[\frac{\mathtt{ca}}{\mathtt{a}}\right], \left[\frac{\mathtt{abc}}{\mathtt{c}}\right] \right\}.$$

  - Taks is to find a **match**: A list of dominos from the collection (with repetitions) to get the same string on top and bottom:

$$\left[\frac{\mathtt{a}}{\mathtt{ab}}\right]\left[\frac{\mathtt{b}}{\mathtt{ca}}\right]\left[\frac{\mathtt{ca}}{\mathtt{a}}\right]\left[\frac{\mathtt{a}}{\mathtt{ab}}\right]\left[\frac{\mathtt{abc}}{\mathtt{c}}\right] \Longleftrightarrow \begin{array}{cccccccc} \mathtt{a} & \mathtt{b} & \mathtt{c} & \mathtt{a} & \mathtt{a} & \mathtt{a} & \mathtt{b} & \mathtt{c} \\ \mathtt{a} & \mathtt{b} & \mathtt{c} & \mathtt{a} & \mathtt{a} & \mathtt{a} & \mathtt{b} & \mathtt{c} \end{array}$$

A Simple Undecidable Problem

- Undecidability is not limited to problems concerning automata.
  - Post Correspondence Problem, or PCP.
- Definition:
  - A domino that has two sides: $\left[\frac{\mathrm{a}}{\mathrm{ab}}\right]$
  - A collection of dominos:

$$\left\{\left[\frac{\mathrm{b}}{\mathrm{ca}}\right], \left[\frac{\mathrm{a}}{\mathrm{ab}}\right], \left[\frac{\mathrm{ca}}{\mathrm{a}}\right], \left[\frac{\mathrm{abc}}{\mathrm{c}}\right]\right\}.$$

  - Taks is to find a **match**: A list of dominos from the collection (with repetitions) to get the same string on top and bottom.
  - Some collections do not contain a match:

$$\left\{\left[\frac{\mathrm{abc}}{\mathrm{ab}}\right], \left[\frac{\mathrm{ca}}{\mathrm{a}}\right], \left[\frac{\mathrm{acc}}{\mathrm{ba}}\right]\right\}$$

A Simple Undecidable Problem

- Undecidability is not limited to problems concerning automata.
- Precise statement:
  - A collection P of dominos: $P = \left\{ \left[\frac{t_1}{b_1}\right], \left[\frac{t_2}{b_2}\right], \ldots, \left[\frac{t_k}{b_k}\right] \right\},$
  - A match: A sequence $i_1, i_2, \ldots, i_l$, where
    $$t_{i_1} t_{i_2} \ldots t_{i_l} = b_{i_1} b_{i_1} \ldots b_{i_l}$$
  - Problem: Determine whether P has a match.

$$PCP = \{\langle P \rangle \mid P \text{ is an instance of the Post Correspondence Problem with a match}\}.$$

A Simple Undecidable Problem

Theorem
PCP is undecidable.

- **Proof Idea**: Reduction from A_{TM} via accepting computation histories.
  - Construct an instance of PCP from any TM M and input w,
  - The match is an accepting computation history for M on w,
    - A match is a simulation of M.
  - Determine a match exists → Determine M accepts w.
  - Three technical details (Modified PCP or MPCP):
    - M on w never moves off the left-hand end of the tape,
    - If w=ε, use ⊔ instead.
    - A match starts with the first domino $\left[\frac{t_1}{b_1}\right]$.

A Simple Undecidable Problem

Theorem
PCP is undecidable.

- **Proof**:
  - Assume TM R decides PCP,
  - Construct TM S that decides A_{TM}.
  - Consider TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$,
  - S constructs an instance of PCP P that has a match iff M accepts w.
  - S first constructs an instance P’ of the MPCP.

A Simple Undecidable Problem

Theorem
PCP is undecidable.

- **Proof**: S first constructs an instance P' of the MPCP:
- Part 1: First configuration.

Put $\left[\frac{\#}{\#q_0w_1w_2\cdots w_n\#}\right]$ into P' as the first domino $\left[\frac{t_1}{b_1}\right]$.

- A match starts with the first configuration (C1) in the accepting computation history for M on w.
- We need more dominos causing next steps in simulation of M.

A Simple Undecidable Problem

Theorem
PCP is undecidable.

- **Proof**: S first constructs an instance P’ of the MPCP:
- Part 2: Move the head to the right.

$$\text{For every } a, b \in \Gamma \text{ and every } q, r \in Q \text{ where } q \neq q_{\text{reject}},$$

$$\text{if } \delta(q, a) = (r, b, \mathrm{R}), \text{ put } \left[\frac{qa}{br}\right] \text{ into } P'.$$

- Part 3: Move the head to the left.

$$\text{For every } a, b, c \in \Gamma \text{ and every } q, r \in Q \text{ where } q \neq q_{\text{reject}},$$

$$\text{if } \delta(q, a) = (r, b, \mathrm{L}), \text{ put } \left[\frac{cqa}{rcb}\right] \text{ into } P'.$$

A Simple Undecidable Problem

Theorem
PCP is undecidable.

- **Proof**: S first constructs an instance P' of the MPCP:
- Part 4: Content of tape not adjacent to the head,

For every $a \in \Gamma$,

$$\text{put } \left[\frac{a}{a}\right] \text{ into } P'.$$

A Simple Undecidable Problem

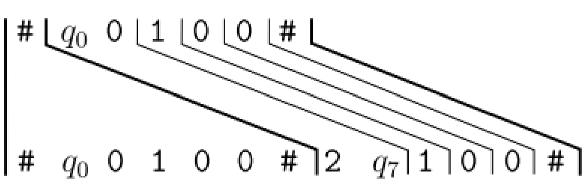
Theorem
PCP is undecidable.

- **Proof**: S first constructs an instance P’ of the MPCP:
- Part 5:
  - Copy # that marks the separation of the configuration,
  - Add a blank at the end to simulate the infinite tape.

$$\text{Put } \left[\frac{\#}{\#}\right] \text{ and } \left[\frac{\#}{\sqcup\#}\right] \text{ into } P'.$$

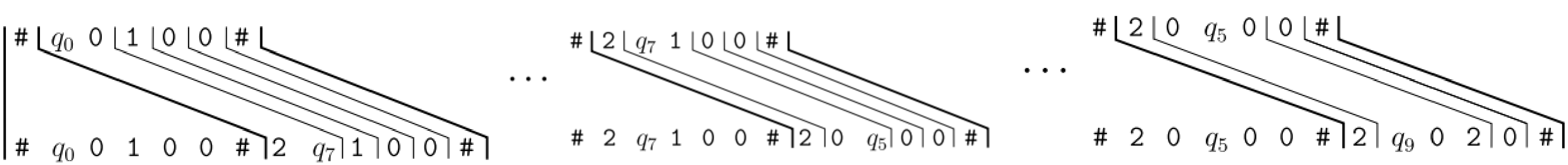
A Simple Undecidable Problem

- **Example:**
  - $\Gamma = \{0,1,2,\sqcup\}$
  - w=0100
  - Start state = q_0
  - $\delta(q_0, 0) = (q_7, 2, R)$
  - Part 1:
    $$\left[\frac{\#}{\#q_0 0100\#}\right] = \left[\frac{t_1}{b_1}\right]$$
  - Part 2:
    $$\left[\frac{q_0 0}{2q_7}\right]$$
  - Part 4:
    $$\left[\frac{0}{0}\right], \left[\frac{1}{1}\right], \left[\frac{2}{2}\right], \text{ and } \left[\frac{\sqcup}{\sqcup}\right]$$



A Simple Undecidable Problem

- **Example:**
  - $\Gamma = \{0,1,2,\sqcup\}$
  - w=0100
  - Start state = q_0
  - $\delta(q_5, 0) = (q_9, 2, L)$
  - Part 1: $$\left[\frac{\#}{\#q_0 0100\#}\right] = \left[\frac{t_1}{b_1}\right]$$
  - Part 2: $$\left[\frac{q_0 0}{2q_7}\right] \quad \left[\frac{q_7 1}{0 q_5}\right] \quad \left[\frac{0 q_5 0}{q_9 0 2}\right], \left[\frac{1 q_5 0}{q_9 1 2}\right], \left[\frac{2 q_5 0}{q_9 2 2}\right], \text{ and } \left[\frac{\sqcup q_5 0}{q_9 \sqcup 2}\right]$$
  - Part 4: $$\left[\frac{0}{0}\right], \left[\frac{1}{1}\right], \left[\frac{2}{2}\right], \text{ and } \left[\frac{\sqcup}{\sqcup}\right]$$



A Simple Undecidable Problem

Theorem
PCP is undecidable.

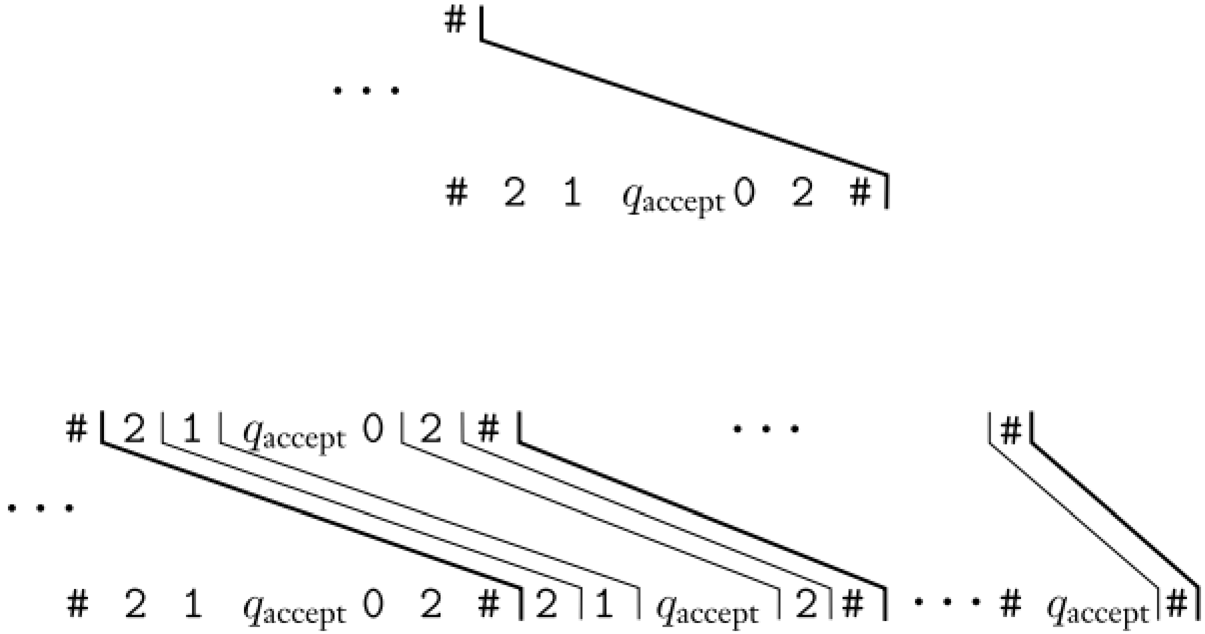
- **Proof**: S first constructs an instance P' of the MPCP:
- If the accept state occurs, we want to let the top of the partial match "**catch up**" with the bottom so that the match is complete.
- Part 6:
  - After halt, the head "eats" adjacent symbols until none are left:

For every $a \in \Gamma$,

$$\text{put } \left[\frac{a\, q_{\text{accept}}}{q_{\text{accept}}}\right] \text{ and } \left[\frac{q_{\text{accept}}\, a}{q_{\text{accept}}}\right] \text{ into } P'.$$

A Simple Undecidable Problem

- **Example:**
  - $\Gamma = \{0,1,2,\sqcup\}$
  - w=0100



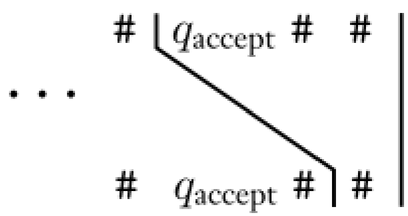
A Simple Undecidable Problem

Theorem
PCP is undecidable.

- **Proof**: S first constructs an instance P' of the MPCP:
- Part 7:
  - Complete the match:

$$\left[\frac{q_{\text{accept}}\#\#}{\#}\right]$$

- **Example**:



A Simple Undecidable Problem

Theorem
PCP is undecidable.

- **Proof**: Convert P' (an instance of MPCP) to P (an instance of PCP):
  - Define star operators:

$$\begin{aligned} \star u &= * u_1 * u_2 * u_3 * \cdots * u_n \\ u\star &= u_1 * u_2 * u_3 * \cdots * u_n * \\ \star u \star &= * u_1 * u_2 * u_3 * \cdots * u_n * . \end{aligned}$$

  - Modify the collection:

Add the extra $*$ at the end of the match

$$\left\{ \left[\frac{t_1}{b_1}\right], \left[\frac{t_2}{b_2}\right], \left[\frac{t_3}{b_3}\right], \ldots, \left[\frac{t_k}{b_k}\right] \right\} \Rightarrow \left\{ \left[\frac{\star t_1}{\star b_1 \star}\right], \left[\frac{\star t_1}{b_1 \star}\right], \left[\frac{\star t_2}{b_2 \star}\right], \left[\frac{\star t_3}{b_3 \star}\right], \ldots, \left[\frac{\star t_k}{b_k \star}\right], \left[\frac{*\Diamond}{\Diamond}\right] \right\}$$

Mapping Reducibility

- Formalize the notion of reducibility,
  - Use reducibility in more refined ways.
  - Prove Turing-nonrecognizability,
  - Useful in complexity theory.

- Mapping reducibility or many-one reducibility.
  - Reduce problem A to B using a mapping reducibility: A **computable function** exists that converts instances of A to instances of B.

Mapping Reducibility

- Definition:

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

- Example:
  - Arithmetic operations on integers,
    - $\langle m, n \rangle \rightarrow \langle m + n \rangle$
  - Transformation of machine descriptions,
    - $\langle M \rangle \rightarrow \langle M' \rangle$, where M' never attempts to move its head off the left-hand end of its tape.

Mapping Reducibility

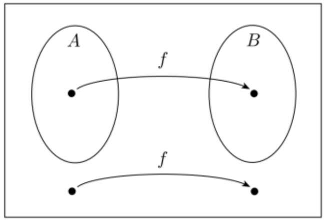
- Definition:

Language A is ***mapping reducible*** to language B, written $A \leq_{\mathrm{m}} B$, if there is a computable function $f\colon \Sigma^* \longrightarrow \Sigma^*$, where for every w,

$$w \in A \Longleftrightarrow f(w) \in B.$$

The function f is called the ***reduction*** from A to B.

- Convert membership testing in A to membership testing in B.
- Illustration of function f reducing A to B:



Mapping Reducibility

Theorem

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

- Proof: Let
  - M be the decider for B and
  - f be the reduction from A to B.
  - A decider N for A:

$N =$ "On input w:

1. Compute $f(w)$.

2. Run M on input $f(w)$ and output whatever M outputs."

Corollary

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

Mapping Reducibility

- **Example**: A mapping reduction from A_{TM} to $HALT_{TM}$.
- $\langle M', w'\rangle = f(\langle M, w\rangle)$, where

$$\langle M, w\rangle \in A_{\mathsf{TM}} \text{ if and only if } \langle M', w'\rangle \in HALT_{\mathsf{TM}}.$$

$F =$ "On input $\langle M, w\rangle$:

1. Construct the following machine M'.
$M' =$ "On input x:
  1. Run M on x.
  2. If M accepts, *accept*.
  3. If M rejects, enter a loop."

2. Output $\langle M', w\rangle$."

- If input is improper output is a string outside B.

Mapping Reducibility

Theorem

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

- Proof: Let
  - M be the recognizer for B and
  - f be the reduction from A to B.
  - A recognizer N for A:

$N =$ "On input w:

1. Compute $f(w)$.
2. Run M on input $f(w)$ and output whatever M outputs."

Corollary

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

Mapping Reducibility

Theorem
EQ_{TM} is neither Turing-recognizable nor co-Turing-recognizable.

- **Proof**:
  - The reduction function f for $A_{TM} \leq_m \overline{EQ_{TM}}$:

$F =$ "On input $\langle M, w\rangle$, where M is a TM and w a string:
  1. Construct the following two machines, M_1 and M_2.
    $M_1 =$ "On any input:
      1. *Reject*."
    $M_2 =$ "On any input:
      1. Run M on w. If it accepts, *accept*."
  2. Output $\langle M_1, M_2\rangle$."

  - If M accepts w, M_2 accepts everything.
    - M_1 and M_2 are not equivalent.
  - If M rejects w, M_2 rejects everything.
    - M_1 and M_2 are equivalent.

Mapping Reducibility

Theorem
EQ_{TM} is neither Turing-recognizable nor co-Turing-recognizable.

- **Proof**:
  - The reduction function g for $A_{TM} \leq_m EQ_{TM}$:

G = "On input $\langle M, w\rangle$, where M is a TM and w a string:
1. Construct the following two machines, M_1 and M_2.
M_1 = "On any input:
1. *Accept*."
M_2 = "On any input:
1. Run M on w.
2. If it accepts, *accept*."
2. Output $\langle M_1, M_2\rangle$."

  - If M accepts w, M_2 accepts everything.
    - M_1 and M_2 are equivalent.
  - If M rejects w, M_2 rejects everything.
    - M_1 and M_2 are not equivalent.