CS310 : Automata Theory 2019

Lecture 33: PCP and its application to CFLs

Instructor: S. Akshay

IITB, India

28-03-2019

Recap

Turing machines and computability

1. Definition of Turing machines: high level and low-level descriptions
2. Variants of Turing machines
3. Decidable and Turing recognizable languages
4. Church-Turing Hypothesis
5. Undecidability and a proof technique by diagonalization
6. Reductions: a powerful way to show undecidability.
7. Rice's theorem, its proof and its applications.
8. Post's Correspondance Problem, its proof and its applications.

Post's correspondance problem

Theorem

The Post's correspondance problem is undecidable.

Proof Idea:

- Encode TM computation histories!

Post's correspondance problem

Theorem

The Post's correspondance problem is undecidable.

Proof Idea:

- Encode TM computation histories!
- Each transition as a domino!

Post's correspondance problem

Theorem

The Post's correspondance problem is undecidable.

Proof Idea:

- Encode TM computation histories!
- Each transition as a domino!
- Simulate the run using the dominos.

Proof of undecidability of PCP:1

Simplifying assumptions

- Assume that the tape of TM is one-way infinite and never attempts to move left off its left-end.

Proof of undecidability of PCP:1

Simplifying assumptions

- Assume that the tape of TM is one-way infinite and never attempts to move left off its left-end.
- If $w = \varepsilon$, then use $\sqcup$ instead of w.

Proof of undecidability of PCP:1

Simplifying assumptions

- Assume that the tape of TM is one-way infinite and never attempts to move left off its left-end.
- If $w = \varepsilon$, then use $\sqcup$ instead of w.
- Modify PCP so that match must start with a given domino, say the first one. Call this MPCP.

Proof of undecidability of PCP:2

We define a reduction from A_{TM} to $(M)PCP$. Let an instance of A_{TM} be

- $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$
- $w = w_1, \ldots w_n$.

Proof of undecidability of PCP:2

We define a reduction from A_{TM} to $(M)PCP$. Let an instance of A_{TM} be

- $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$
- $w = w_1, \ldots w_n$.

We build instance P' of MPCP in several steps:

Proof of undecidability of PCP:2

We define a reduction from A_{TM} to $(M)PCP$. Let an instance of A_{TM} be

- $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$
- $w = w_1, \ldots w_n$.

We build instance P' of MPCP in several steps:

Step 1: fix first domino in P'

$$\left[\frac{\#}{\# q_0 w_1 \cdots w_n \#}\right]$$

Proof of undecidability of PCP:2

We define a reduction from A_{TM} to $(M)PCP$. Let an instance of A_{TM} be

- $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$
- $w = w_1, \ldots w_n$.

We build instance P' of MPCP in several steps:

Step 1: fix first domino in P'

$$\left[\frac{\#}{\# q_0 w_1 \cdots w_n \#}\right]$$

Because we are reducing to MPCP, the match must start with this domino!

Proof of undecidability of PCP:2

We define a reduction from A_{TM} to $(M)PCP$. Let an instance of A_{TM} be

- $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$
- $w = w_1, \ldots w_n$.

We build instance P' of MPCP in several steps:

Step 1: fix first domino in P'

$$\left[\frac{\#}{\# q_0 w_1 \cdots w_n \#}\right]$$

Because we are reducing to MPCP, the match must start with this domino!How do we proceed?

Proof of undecidability of PCP:3

Step 2: encode transitions of TM into dominos!

Proof of undecidability of PCP:3

Step 2: encode transitions of TM into dominos!

For every $a, b, c \in \Gamma$ and every $q, q' \in Q$, $q \neq q_{rej}$,

- if $\delta(q, a) = (q', b, R)$ then add domino to P':

$$\left[\frac{qa}{bq'}\right]$$

Proof of undecidability of PCP:3

Step 2: encode transitions of TM into dominos!

For every $a, b, c \in \Gamma$ and every $q, q' \in Q$, $q \neq q_{rej}$,

- if $\delta(q, a) = (q', b, R)$ then add domino to P':

$$\left[\frac{qa}{bq'}\right]$$

- if $\delta(q, a) = (q', b, L)$ then add domino to P':

$$\left[\frac{cqa}{q'cb}\right]$$

Proof of undecidability of PCP:3

Step 2: encode transitions of TM into dominos!

For every $a, b, c \in \Gamma$ and every $q, q' \in Q$, $q \neq q_{rej}$,

- if $\delta(q, a) = (q', b, R)$ then add domino to P':

$$\left[\frac{qa}{bq'}\right]$$

- if $\delta(q, a) = (q', b, L)$ then add domino to P':

$$\left[\frac{cqa}{q'cb}\right]$$

- add all dominos (i.e, for all $a \in \Gamma \cup \{\#\}$) to P':

$$\left[\frac{a}{a}\right]$$

Proof of undecidability of PCP:3

Step 2: encode transitions of TM into dominos!

For every $a, b, c \in \Gamma$ and every $q, q' \in Q$, $q \neq q_{rej}$,

- if $\delta(q, a) = (q', b, R)$ then add domino to P':

$$\left[\frac{qa}{bq'}\right]$$

- if $\delta(q, a) = (q', b, L)$ then add domino to P':

$$\left[\frac{cqa}{q'cb}\right]$$

- add all dominos (i.e, for all $a \in \Gamma \cup \{\#\}$) to P':

$$\left[\frac{a}{a}\right] \text{ and } \left[\frac{\#}{\sqcup\#}\right]$$

to model adding new blanks on right, when needed

Proof of undecidability of PCP:4

Step 3: acceptance into eating dominos

Proof of undecidability of PCP:4

Step 3: acceptance into eating dominos

For every $a \in \Gamma$, we add foll dominos to P':

$$\left[\frac{q_{acc}\, a}{q_{acc}}\right], \left[\frac{a q_{acc}}{q_{acc}}\right]$$

Proof of undecidability of PCP:4

Step 3: acceptance into eating dominos

For every $a \in \Gamma$, we add foll dominos to P':

$$\left[\frac{q_{acc}a}{q_{acc}}\right], \left[\frac{aq_{acc}}{q_{acc}}\right]$$

Exercise: What happens in the previous example if we reach:

$$\ldots \left[\frac{\#}{\#\ 2\ 1\ q_{acc}\ 0\ 2\ \#}\right]$$

Proof of undecidability of PCP:4

Step 3: acceptance into eating dominos

For every $a \in \Gamma$, we add foll dominos to P':

$$\left[\frac{q_{acc}a}{q_{acc}}\right], \left[\frac{aq_{acc}}{q_{acc}}\right]$$

Step 4: complete the match

Add:

$$\left[\frac{q_{acc}\#\#}{\#}\right]$$

Proof of undecidability of PCP:4

Step 3: acceptance into eating dominos

For every $a \in \Gamma$, we add foll dominos to P':

$$\left[\frac{q_{acc}a}{q_{acc}}\right], \left[\frac{aq_{acc}}{q_{acc}}\right]$$

Step 4: complete the match

Add:

$$\left[\frac{q_{acc}\#\#}{\#}\right]$$

This completes the reduction

- i.e., map from instance of A_{TM} to instance of $MPCP$ s.t.,
- M acc w iff P' gives a solution to MPCP problem.

Proof of undecidability of PCP:4

Step 3: acceptance into eating dominos

For every $a \in \Gamma$, we add foll dominos to P':

$$\left[\frac{q_{acc}a}{q_{acc}}\right], \left[\frac{aq_{acc}}{q_{acc}}\right]$$

Step 4: complete the match

Add:

$$\left[\frac{q_{acc}\#\#}{\#}\right]$$

This completes the reduction

- i.e., map from instance of A_{TM} to instance of $MPCP$ s.t.,
- M acc w iff P' gives a solution to MPCP problem.

Does this also give a reduction to PCP?

Proof of undecidability of PCP:5

Reduction from MPCP to PCP!

For every domino $d = \left[\frac{a_1 \ldots a_r}{b_1 \ldots b_s}\right]$ of P'

Proof of undecidability of PCP:5

Reduction from MPCP to PCP!

For every domino $d = \left[\frac{a_1 \ldots a_r}{b_1 \ldots b_s}\right]$ of P'

- for every d in P', we add in P

$$\left[\frac{*a_1 * a_2 \ldots * a_r}{b_1 * b_2 \ldots * b_s *}\right]$$

This completes the reduction and the proof!

Proof of undecidability of PCP:5

Reduction from MPCP to PCP!

For every domino $d = \left[\frac{a_1 \ldots a_r}{b_1 \ldots b_s}\right]$ of P'

- for every d in P', we add in P

$$\left[\frac{*a_1 * a_2 \ldots * a_r}{b_1 * b_2 \ldots * b_s *}\right]$$

- if d is the first one, we additionally add in P,

$$\left[\frac{*a_1 * a_2 \ldots * a_r}{*b_1 * b_2 \ldots * b_s *}\right]$$

This completes the reduction and the proof!

Proof of undecidability of PCP:5

Reduction from MPCP to PCP!

For every domino $d = \left[\frac{a_1 \ldots a_r}{b_1 \ldots b_s}\right]$ of P'

- for every d in P', we add in P

$$\left[\frac{*a_1 * a_2 \ldots * a_r}{b_1 * b_2 \ldots * b_s *}\right]$$

- if d is the first one, we additionally add in P,

$$\left[\frac{*a_1 * a_2 \ldots * a_r}{*b_1 * b_2 \ldots * b_s *}\right]$$

Also to finish the match, add in P,

$$\left[\frac{*\diamond}{\diamond}\right]$$

This completes the reduction and the proof!

Another simple problem

Thus, the string matching problem (PCP) is undecidable!

Given two lists $A = \{s_1, \dots s_n\}$ and $B = \{t_1, \dots, t_n\}$, over the same alphabet,

- does there exist a finite sequence $1 \leq i_1, \dots, i_m \leq n$ such that

$$s_{i_1} \dots s_{i_m} = t_{i_1} \dots t_{i_m}$$

Another simple problem

Thus, the string matching problem (PCP) is undecidable!

Given two lists $A = \{s_1, \dots s_n\}$ and $B = \{t_1, \dots, t_n\}$, over the same alphabet,

- does there exist a finite sequence $1 \leq i_1, \dots, i_m \leq n$ such that

$$s_{i_1} \dots s_{i_m} = t_{i_1} \dots t_{i_m}$$

A completely different yet natural problem

Another simple problem

Thus, the string matching problem (PCP) is undecidable!

Given two lists $A = \{s_1, \ldots s_n\}$ and $B = \{t_1, \ldots, t_n\}$, over the same alphabet,

- does there exist a finite sequence $1 \leq i_1, \ldots, i_m \leq n$ such that

$$s_{i_1} \ldots s_{i_m} = t_{i_1} \ldots t_{i_m}$$

A completely different yet natural problem

Is a context-free grammar (CFG) ambiguous?

Undecidability of Ambiguity for CFG's

- Reduction from PCP to this problem
- Then, if there is an algorithm for this problem, it will give an algorithm to decide PCP, a contradiction!

Undecidability of Ambiguity for CFG's

- Reduction from PCP to this problem
- Then, if there is an algorithm for this problem, it will give an algorithm to decide PCP, a contradiction!

Given list $A = \{s_1, \ldots s_n\}$ construct CFG G_A, with single variable A and terminals: Σ, set of distinct index symbols $a_1, \ldots, a_n$

$$A \rightarrow s_1 A a_1 \mid s_2 A a_2 \mid \ldots \mid s_n A a_n \mid s_1 a_1 \mid \ldots \mid s_n a_n$$

Undecidability of Ambiguity for CFG's

- Reduction from PCP to this problem
- Then, if there is an algorithm for this problem, it will give an algorithm to decide PCP, a contradiction!

Given list $A = \{s_1, \ldots s_n\}$ construct CFG G_A, with single variable A and terminals: Σ, set of distinct index symbols $a_1, \ldots, a_n$

$$A \to s_1 A a_1 \mid s_2 A a_2 \mid \ldots \mid s_n A a_n \mid s_1 a_1 \mid \ldots \mid s_n a_n$$

- What are the terminal strings of G_A?

Undecidability of Ambiguity for CFG's

- Reduction from PCP to this problem
- Then, if there is an algorithm for this problem, it will give an algorithm to decide PCP, a contradiction!

Given list $A = \{s_1, \dots s_n\}$ construct CFG G_A, with single variable A and terminals: Σ, set of distinct index symbols $a_1, \dots, a_n$

$$A \to s_1 A a_1 \mid s_2 A a_2 \mid \dots \mid s_n A a_n \mid s_1 a_1 \mid \dots \mid s_n a_n$$

- What are the terminal strings of G_A?
- Is G_A ambiguous? That is, for any terminal string, how many derivations does it have?
- The index symbol at the end of string determines (uniquely) which production was used at a step.

Undecidability of Ambiguity for CFG's

Given list $B = \{t_1, \ldots, t_n\}$ construct CFG G_B, with single variable B and terminals: Σ, set of distinct index symbols $a_1, \ldots, a_n$

$$B \rightarrow t_1 B a_1 \mid t_2 B a_2 \mid \ldots \mid t_n B a_n \mid t_1 a_1 \mid \ldots \mid t_n a_n$$

Same properties hold for G_B (as for G_A)

Undecidability of Ambiguity for CFG's

Given list $B = \{t_1, \ldots, t_n\}$ construct CFG G_B, with single variable B and terminals: Σ, set of distinct index symbols $a_1, \ldots, a_n$

$$B \to t_1 B a_1 \mid t_2 B a_2 \mid \ldots \mid t_n B a_n \mid t_1 a_1 \mid \ldots \mid t_n a_n$$

Same properties hold for G_B (as for G_A)

Now, given an instance of PCP, i.e., $A = \{s_1, \ldots, s_n\}$ and $B = \{t_1, \ldots, t_n\}$, construct CFG G_{AB}

- Variables are A, B, S, S is start symbol
- Production $S \to A \mid B$
- All productions of G_A, G_B

Undecidability of Ambiguity for CFG's

Given list $B = \{t_1, \ldots, t_n\}$ construct CFG G_B, with single variable B and terminals: Σ, set of distinct index symbols $a_1, \ldots, a_n$

$$B \to t_1 B a_1 \mid t_2 B a_2 \mid \ldots \mid t_n B a_n \mid t_1 a_1 \mid \ldots \mid t_n a_n$$

Same properties hold for G_B (as for G_A)

Now, given an instance of PCP, i.e., $A = \{s_1, \ldots, s_n\}$ and $B = \{t_1, \ldots, t_n\}$, construct CFG G_{AB}

- Variables are A, B, S, S is start symbol
- Production $S \to A \mid B$
- All productions of G_A, G_B

Claim: G_{AB} is ambiguous iff instance (A, B) of PCP has a solution

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem:
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \rightarrow G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\implies$ Spse $i_1, \ldots, i_m$ is a soln to PCP.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\implies$ Spse $i_1, \ldots, i_m$ is a soln to PCP.
  - This implies $s_{i_1} \ldots s_{i_m} = t_{i_1} \ldots t_{i_m}$.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\implies$ Spse $i_1, \ldots, i_m$ is a soln to PCP.
  - This implies $s_{i_1} \ldots s_{i_m} = t_{i_1} \ldots t_{i_m}$.
  - Can you give a string which has two (distinct, leftmost) derivations in G_{AB}?

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\implies$ Spse $i_1, \ldots, i_m$ is a soln to PCP.
  - This implies $s_{i_1} \ldots s_{i_m} = t_{i_1} \ldots t_{i_m}$.
  - Can you give a string which has two (distinct, leftmost) derivations in G_{AB}?
  - Thus, G_{AB} is ambiguous.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\Longleftarrow$ Spse G_{AB} has two leftmost derivations.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\Longleftarrow$ Spse G_{AB} has two leftmost derivations.
  - One must begin with $S \implies A$ and other with $S \implies B$ and derive same string (why?)

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \rightarrow G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\Longleftarrow$ Spse G_{AB} has two leftmost derivations.
  - One must begin with $S \implies A$ and other with $S \implies B$ and derive same string (why?)
  - The tail of this string has some indices $a_{i_m} \ldots a_{i_1}$ for some $m \geq 1$.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- Show that this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
  - Proof: $\Longleftarrow$ Spse G_{AB} has two leftmost derivations.
  - One must begin with $S \implies A$ and other with $S \implies B$ and derive same string (why?)
  - The tail of this string has some indices $a_{i_m} \dots a_{i_1}$ for some $m \geq 1$.
  - This is a solution to PCP instance, since what precedes is both $s_{i_1} \dots s_{i_m}$ and $t_{i_1} \dots t_{i_m}$.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \rightarrow G_{AB}$
- this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
- Thus, undecidability of PCP implies undecidability of checking ambiguity of CFG.

Undecidability of Ambiguity for CFG's

Theorem: Checking if a CFG is ambiguous is undecidable

Proof:

- Map instance of PCP to instance of this problem: $(A, B) \to G_{AB}$
- this is a reduction, i.e., instance (A, B) of PCP has a solution iff G_{AB} is ambiguous.
- Thus, undecidability of PCP implies undecidability of checking ambiguity of CFG.
  - i.e., if we had an algorithm to decide unambiguity of CFG, we could apply the reduction and obtain an algorithm to decide PCP.

Undecidable properties about CFLs

- Let $L_A = L(G_A)$ be the CFL accepting G_A.

Undecidable properties about CFLs

- Let $L_A = L(G_A)$ be the CFL accepting G_A.
- What about $\overline{L_A}$? Language of strings over $\Sigma \cup \{a_1, \ldots a_n\}$ that are not in L_A.

Undecidable properties about CFLs

- Let $L_A = L(G_A)$ be the CFL accepting G_A.
- What about $\overline{L_A}$? Language of strings over $\Sigma \cup \{a_1, \ldots a_n\}$ that are not in L_A.

Theorem: $\overline{L_A}$ is context-free.

Proof: Define a deterministic PDA.

Undecidable properties about CFLs

- Let $L_A = L(G_A)$ be the CFL accepting G_A.
- What about $\overline{L_A}$? Language of strings over $\Sigma \cup \{a_1, \ldots a_n\}$ that are not in L_A.

Theorem: $\overline{L_A}$ is context-free.

Proof: Define a deterministic PDA. Home-work!

Undecidable properties about CFLs

- Let $L_A = L(G_A)$ be the CFL accepting G_A.
- What about $\overline{L_A}$? Language of strings over $\Sigma \cup \{a_1, \ldots a_n\}$ that are not in L_A.

Theorem: $\overline{L_A}$ is context-free.

Proof: Define a deterministic PDA.

Let G_1, G_2 be CFGs and R be a regular expression

- Is $L(G_1) \cap L(G_2) \neq \emptyset$?
- Is $L(G_1) \cap L(G_2) = \emptyset$?
- Is $L(G) = L(R)$?

Undecidable properties about CFLs

- Let $L_A = L(G_A)$ be the CFL accepting G_A.
- What about $\overline{L_A}$? Language of strings over $\Sigma \cup \{a_1, \ldots a_n\}$ that are not in L_A.

Theorem: $\overline{L_A}$ is context-free.

Proof: Define a deterministic PDA.

Let G_1, G_2 be CFGs and R be a regular expression

- Is $L(G_1) \cap L(G_2) \neq \emptyset$? Take $L(G_1) = L_A$, $L(G_2) = L_B$
- Is $L(G_1) \cap L(G_2) = \emptyset$?
- Is $L(G) = L(R)$?