Lec 12. Pumping Lemma for Context-Free Languages

Eunjung Kim

Pumping lemma

Pumping lemma for CFL

Let A be a context-free language. Then there exists a number p (the pumping length) such that any string $w \in A$ of length at least p, w can be written as $w = uvxyz$ such that the following holds:

1. $|vy| \geq 1$,
2. $|vxy| \leq p$,
3. $uv^ixy^iz \in A$ for every $i \geq 0$.

Pumping lemma

Proof idea: For a sufficiently long string *w* and its parse tree, some variable is used at least twice.

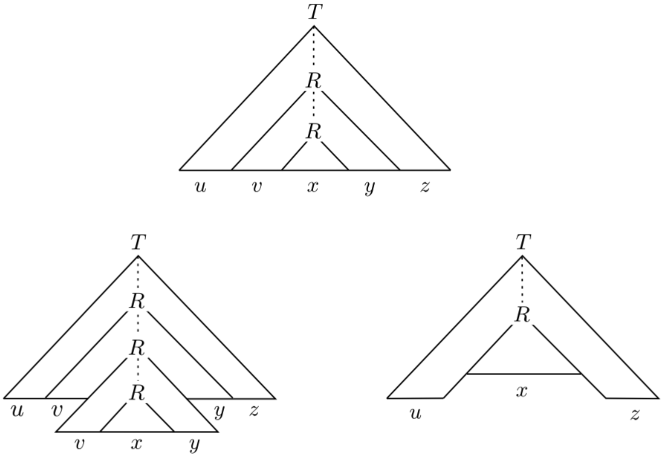

Figure 2.35 from Sipser 2012.

Pumping lemma for CFL, proof

There exists a context-free grammar $G = (V, \Sigma, S, R)$ with $L(M) = A$.

- Let b be the max number of symbols in the rhs of a rule.
- In any parse tree in this grammar, an internal node has $\leq b$ children.
- Any parse tree has $\leq b^h$ leaves, where h is the height of a parse tree in G.
- Let $p := b^{|V|+1}$.
- If $w \in A$ has length at least p, then its parse tree has $\geq p = b^{|V|+1}$ leaves, and height at least $|V| + 1$.
- Choose a parse tree τ yielding w with minimum number of nodes.
- Take a longest root-to-leaf path Q in τ; has length at least $|V| + 1$.
- Q has at least $|V| + 2$ nodes; only the last node is a terminal, the other $\geq |V| + 1$ nodes are variables.

Pumping lemma for CFL, proof

Let X be a variable which occurs twice in the last $|V| + 1$ variables on Q. Rewrite $w = uvxyz$, where vxy is the yield of the first X, and x is the yield of the second X.

1. $uv^ixy^iz \in A$: replacing the subtree rooted at the second X by the one rooted at the first X (or vice versa)
2. $|vy| \geq 1$: if $vy = \epsilon$, then replacing the subtree rooted at the first X by the subtree rooted at the second X leads to a parse tree with strictly smaller number of nodes. Contradicts the choice of τ.
3. $|vxy| \leq p$: the subtree rooted at the first X has height at most $|V| + 1$ by the choice of X. It has $\leq b^{|V|+1}$ leaves, thus its yield vxy has length $\leq b^{|V|+1} = p$.

Using Pumping lemma for proving non-CFL

Pumping lemma for CFL, restated

Let A be a context-free language. Then

1. there exists p such that
2. for any string $w \in A$ of length at least p,
3. there exists a rewriting of w as $w = uvxyz$ with $|vy| \geq 1$ and $|vxy| \leq p$ such that
4. for any $i \geq 0$, it holds that $uv^ixy^iz \in A$.

Using Pumping lemma for proving non-CFL

We use the contraposition of Pumping lemma for proving A isn't CFL.

Syntax

1. For every positive number p, ("$\forall p$")
2. there exists $w \in A$ of length at least p such that ("$\exists w \in A$")
3. for every split $w = uvxyz$ with $|vy| \geq 1$ and $|vxy| \leq p$
4. there exists $i \geq 0$ with $uv^i xy^i z \notin A$ ("$\exists i$").

If one can establish the above for a language A, then by (the contrapositive of) Pumping Lemma we prove that A is not CFL.

Using Pumping lemma for proving non-CFL

1. $B = \{a^i b^j c^k \mid 0 \leq i \leq j \leq k\}$
2. $C = \{ww \mid w \in \{0, 1\}^*\}$

$$B = \{a^i b^j c^k \mid 0 \le i \le j \le k\}$$

$C = \{ww \mid w \in \{0, 1\}^*\}$

Membership test for CFL

Polynomial-time algorithm for membership test

1. Convert a representation (PDA, CFG) of CFL A into CFG of Chomsky Normal Form.
2. CYK (Cocke–Younger–Kasami) algorithm: fill in a $n \times n$ table to test if $w \in A$ for a string w of length $n \geq 1$.

Chomsky Normal Form

Chomsky Normal Form

A context-free grammar (V, Σ, S, R) is in Chomsky Normal Form if every rule is one of the following form

1. $X \to YZ$ for variables $X, Y, Z \in V$.
2. $X \to a$ for a terminal $a \in \Sigma$.
3. $S \to \epsilon$.

Remark: If $\epsilon \notin A$ for CFL A, then there is no ϵ-rule (i.e. a production rule whose body is ϵ). If $\epsilon \in A$, then $S \to \epsilon$ must be one of the production rule, and there is no other ϵ-rule.

Remark: one can convert any CFG G into Chomsky Normal Form in time $O(|G|^2)$.

CYK algorithm in $O(n^3)$-time

INPUT: a grammar $G = (V, \Sigma, S, R)$ in Chomsky Normal Form, a string $w \in \Sigma^*$.

QUESTION: is $w \in L(G)$?

CYK algorithm in $O(n^3)$-time

INPUT: a grammar $G = (V, \Sigma, S, R)$ in Chomsky Normal Form, a string $w \in \Sigma^*$.

QUESTION: is $w \in L(G)$?

Observation:
Suppose $X \Rightarrow^* w$. Then in any derivation of w, the first production rule applied is

- of the form $X \to a$ for some $a \in \Sigma_\epsilon$ if $|w| \leq 1$, and
- of the form $X \to YZ$ for $X, Y, Z \in V$ if $|w| \geq 2$.

CYK algorithm in $O(n^3)$-time

INPUT: a grammar $G = (V, \Sigma, S, R)$ in Chomsky Normal Form, a string $w \in \Sigma^*$.

QUESTION: is $w \in L(G)$?

Observation:
Suppose $X \Rightarrow^* w$. Then in any derivation of w, the first production rule applied is

- of the form $X \rightarrow a$ for some $a \in \Sigma_\epsilon$ if $|w| \leq 1$, and
- of the form $X \rightarrow YZ$ for $X, Y, Z \in V$ if $|w| \geq 2$.

Observation: if $w = \epsilon$, then $w \in L(G)$ if and only if there is a rule $S \rightarrow \epsilon$.

CYK algorithm in $O(n^3)$-time

We may assume that $|w| \geq 1$. Let $w = w_1 \cdot w_n$ for $w_i \in \Sigma$.

Tabulation

For each pair i, j with $1 \leq i \leq j \leq n$, we compute the set $W_{i,j} \subseteq V$ of variables which generates the substring $w[i, j] := w_i \cdots w_j$ of w. That is,

$$W_{ij,} = \{X \in V : X \Rightarrow_G^* w[i, j]\}.$$

CYK ALGORITHM IN $O(n^3)$-TIME

We may assume that $|w| \geq 1$. Let $w = w_1 \cdot w_n$ for $w_i \in \Sigma$.

TABULATION

For each pair i, j with $1 \leq i \leq j \leq n$, we compute the set $W_{i,j} \subseteq V$ of variables which generates the substring $w[i, j] := w_i \cdots w_j$ of w. That is,

$$W_{ij,} = \{X \in V : X \Rightarrow_G^* w[i, j]\}.$$

- For $i = j$: $X \in W_{i,i}$ if and only if there is a rule $X \rightarrow w_i$ in G.

CYK ALGORITHM IN $O(n^3)$-TIME

We may assume that $|w| \geq 1$. Let $w = w_1 \cdot w_n$ for $w_i \in \Sigma$.

TABULATION

For each pair i, j with $1 \leq i \leq j \leq n$, we compute the set $W_{i,j} \subseteq V$ of variables which generates the substring $w[i,j] := w_i \cdots w_j$ of w. That is,

$$W_{ij,} = \{X \in V : X \Rightarrow^*_G w[i,j]\}.$$

- For $i = j$: $X \in W_{i,i}$ if and only if there is a rule $X \to w_i$ in G.
- For $i < j$: $X \Rightarrow^* w[i,j]$ (i.e. $X \in W_{i,k}$) if and only if
  1. there is a rule $X \to YZ$ in G such that
  2. $Y \Rightarrow^* w[i,k]$ and $Z \Rightarrow^* w[k+1,j]$ for some k with $i \leq k < j$, or equivalently

$$Y \in W_{i,k} \qquad \text{and} \quad Z \in W_{k+1,j}$$

CYK ALGORITHM IN $O(n^3)$-TIME

We may assume that $|w| \geq 1$. Let $w = w_1 \cdot w_n$ for $w_i \in \Sigma$.

TABULATION

For each pair i, j with $1 \leq i \leq j \leq n$, we compute the set $W_{i,j} \subseteq V$ of variables which generates the substring $w[i,j] := w_i \cdots w_j$ of w. That is,

$$W_{ij,} = \{X \in V : X \Rightarrow_G^* w[i,j]\}.$$

- For $i = j$: $X \in W_{i,i}$ if and only if there is a rule $X \to w_i$ in G.
- For $i < j$: $X \Rightarrow^* w[i,j]$ (i.e. $X \in W_{i,k}$) if and only if
  1. there is a rule $X \to YZ$ in G such that
  2. $Y \Rightarrow^* w[i,k]$ and $Z \Rightarrow^* w[k+1,j]$ for some k with $i \leq k < j$, or equivalently

$$Y \in W_{i,k} \qquad \text{and} \quad Z \in W_{k+1,j}$$

- Compute the set of variables W_{ij} in a non-decreasing order of $j - i$.

CYK algorithm, example

Consider the following rules of a grammar G in CNF:

- $S \to AB \mid BC$
- $A \to BA \mid a$
- $B \to CC \mid b$
- $C \to AB \mid a$

We want to check if $baaba \in L(G)$.

Testing emptiness of CFL A

Generating variable

Let $G = (V, \Sigma, S, R)$ be a context-free grammar.

We say that a symbol $\gamma \in V \cup \Sigma \cup \{\epsilon\}$ is generating if there is a string $w \in \Sigma^*$ such that $\gamma \Rightarrow_G^* w$.

Note that $L(G) \neq \emptyset$ if and only if S is generating.

Algorithm: we compute the set of generating symbols by induction on the length of a shortest derivation of a string in Σ^*.

1. Base: mark each symbol $\Sigma \cup \{\epsilon\}$ as "generating" (by length-0 derivation).
2. Induction: mark a (variable) symbol X as "generating" if there is a rule $X \rightarrow \alpha$ with $\alpha \in (V \cup \Sigma)^*$ such that all symbols in α are generating (of max shortest derivation length at most $n - 1$.

Testing emptiness of CFL A

Generating variable

Apply the above procedure exhaustively. A symbol X (in V) is marked as "generating" if and only if X is indeed generating in G.

Forward implication clear.
Backward implication: think of a parse tree if $X \Rightarrow^* w$ for some $w \in \Sigma^*$.