Chapter 17: Context-Free Languages *

Peter Cappello
Department of Computer Science
University of California, Santa Barbara
Santa Barbara, CA 93106
cappello@cs.ucsb.edu

- Please read the corresponding chapter before attending this lecture.
- These notes are not intended to be complete. They are supplemented with figures, and material that arises during the lecture period in response to questions.

*Based on **Theory of Computing**, 2nd Ed., D. Cohen, John Wiley & Sons, Inc.

Closure Properties

Theorem: CFLs are closed under union

If L_1 and L_2 are CFLs, then $L_1 \cup L_2$ is a CFL.

Proof

1. Let L_1 and L_2 be generated by the CFG, $G_1 = (V_1, T_1, P_1, S_1)$ and $G_2 = (V_2, T_2, P_2, S_2)$, respectively.
2. Without loss of generality, subscript each nonterminal of G_1 with a 1, and each nonterminal of G_2 with a 2 (so that $V_1 \cap V_2 = \emptyset$).
3. Define the CFG, G, that generates $L_1 \cup L_2$ as follows:

   $G = (V_1 \cup V_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S \rightarrow S_1 \mid S_2\}, S)$.

4. A derivation starts with either $S \Rightarrow S_1$ or $S \Rightarrow S_2$.
5. Subsequent steps use productions entirely from G_1 or entirely from G_2.
6. Each word generated thus is either a word in L_1 or a word in L_2.

Example

- Let L_1 be PALINDROME, defined by:

  $S \rightarrow aSa \mid bSb \mid a \mid b \mid \Lambda$

- Let L_2 be $\{a^n b^n | n \geq 0\}$ defined by:

  $S \rightarrow aSb \mid \Lambda$

- Then the union language is defined by:

  $S \rightarrow S_1 \mid S_2$

  $S_1 \rightarrow aS_1a \mid bS_1b \mid a \mid b \mid \Lambda$

  $S_2 \rightarrow aS_2b \mid \Lambda$

Theorem: CFLs are closed under concatenation

If L_1 and L_2 are CFLs, then L_1L_2 is a CFL.

Proof

1. Let L_1 and L_2 be generated by the CFG, $G_1 = (V_1, T_1, P_1, S_1)$ and $G_2 = (V_2, T_2, P_2, S_2)$, respectively.
2. Without loss of generality, subscript each nonterminal of G_1 with a 1, and each nonterminal of G_2 with a 2 (so that $V_1 \cap V_2 = \emptyset$).
3. Define the CFG, G, that generates L_1L_2 as follows:

   $G = (V_1 \cup V_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S \rightarrow S_1S_2\}, S)$.
4. Each word generated thus is a word in L_1 followed by a word in L_2.

Example

- Let L_1 be PALINDROME, defined by:

  $S \to aSa \mid bSb \mid a \mid b \mid \Lambda$

- Let L_2 be $\{a^n b^n | n \geq 0\}$ defined by:

  $S \to aSb \mid \Lambda$

- Then the concatenation language is defined by:

  $S \to S_1 S_2$

  $S_1 \to aS_1a \mid bS_1b \mid a \mid b \mid \Lambda$

  $S_2 \to aS_2b \mid \Lambda$

Theorem: CFLs are closed under Kleene star

If L_1 is a CFL, then L_1^* is a CFL.

Proof

1. Let L_1 be generated by the CFG, $G_1 = (V_1, T_1, P_1, S_1)$.
2. Without loss of generality, subscript each nonterminal of G_1 with a 1.
3. Define the CFG, G, that generates L_1^* as follows:
   $G = (V_1 \cup \{S\}, T_1, P_1 \cup \{S \rightarrow S_1 S \mid \Lambda\}, S)$.
4. Each word generated is either Λ or some sequence of words in L_1.
5. Every word in L_1^* (i.e., some sequence of 0 or more words in L_1) can be generated by G.

Example

- Let L_1 be $\{a^n b^n | n \geq 0\}$ defined by:

  $S \rightarrow aSb \mid \Lambda$

- Then L_1^* is generated by:

  $S \rightarrow S_1 S \mid \Lambda$

  $S_1 \rightarrow aS_1 b \mid \Lambda$

None of these example grammars is necessarily the most *compact* CFG for the language it generates.

Intersection and Complement

Theorem: CFLs are not closed under intersection

If L_1 and L_2 are CFLs, then $L_1 \cap L_2$ may not be a CFL.

Proof

1. $L_1 = \{a^n b^n a^m \mid n, m \geq 0\}$ is generated by the following CFG:

   $S \rightarrow XA$

   $X \rightarrow aXb \mid \Lambda$

   $A \rightarrow Aa \mid \Lambda$

2. $L_2 = \{a^n b^m a^m \mid n, m \geq 0\}$ is generated by the following CFG:

   $S \rightarrow AX$

$X \to aXb \mid \Lambda$

$A \to Aa \mid \Lambda$

3. $L_1 \cap L_2 = \{a^n b^n a^n \mid n \geq 0\}$, which is known not to be a CFL (pumping lemma).

Theorem: CFLs are not closed under complement

If L_1 is a CFL, then $\overline{L_1}$ may not be a CFL.

Proof

They are closed under union. If they are closed under complement, then they are closed under intersection, which is false.

More formally,

1. Assume the complement of every CFL is a CFL.
2. Let L_1 and L_2 be 2 CFLs.
3. Since CFLs are close under union, and we are assuming they are closed under complement,
$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$
is a CFL.

4. However, we know there are CFLs whose intersection is not a CFL.
5. Therefore, our assumption that CFLs are closed under complement is false.

Example

This does *not mean* that the complement of a CFL is *never* a CFL.

- Let $L_1 = \{a^n b^n a^n | n \geq 0\}$, which is not a CFL.
- $\overline{L_1}$ is a CFL.
- We show this by constructing it as the union of 5 CFLs.
  - $Mpq = (a^+)(a^n b^n)(a^+) = \{a^p b^q a^r \mid p > q\}$
  - $Mqp = (a^n b^n)(b^+)(a^+) = \{a^p b^q a^r \mid p < q\}$
  - $Mqr = (a^+)(b^+)(b^n a^n) = \{a^p b^q a^r \mid q > r\}$
  - $Mqr = (a^+)(b^n a^n)(a^+) = \{a^p b^q a^r \mid q < r\}$
  - $M = \overline{a^+ b^+ a^+}$ = all words not of the form $a^p b^q a^r$.

  Let $L = M \cup Mpq \cup Mqp \cup Mqr \cup Mqr$.
- Since $M \subseteq L$, $\overline{L}$ contains only words of the form $a^p b^q a^r$.

- $\overline{L}$ cannot contain words of the form $a^p b^q a^r$, where $p < q$.
- $\overline{L}$ cannot contain words of the form $a^p b^q a^r$, where $p > q$.
- Therefore $\overline{L}$ only contains words of the form $a^p b^q a^r$, where $p = q$.
- $\overline{L}$ cannot contain words of the form $a^p b^q a^r$, where $q < r$.
- $\overline{L}$ cannot contain words of the form $a^p b^q a^r$, where $q > r$.
- Therefore $\overline{L}$ only contains words of the form $a^p b^q a^r$, where $q = r$.
- Since $p = q$ and $q = r$, $\overline{L}$ contains words of the form $a^n b^n a^n$, which is not context-free.

Theorem: The intersection of a CFL and an RL is a CFL.

If L_1 is a CFL and L_2 is regular, then $L_1 \cap L_2$ is a CFL.

Proof

1. We do this by constructing a PDA I to accept the intersection that is based on a PDA A for L_1 and a FA F for L_2.
2. Convert A, if necessary, so that all input is read before accepting.
3. Construct a set Y of all A's states $y_1, y_2, \ldots$, and a set X of all F's states $x_1, x_2, \ldots$.
4. Construct $\{(y, x) \mid \forall y \in Y, \forall x \in X\}$.
5. The start state of I is (y_0, x_0), where y_0 is the label of A's start state, and x_0 is F's initial state.

6. Regarding the next state function, the x component changes only when the PDA is in a READ state:

   - If in (y_i, x_j) and y_i is not a READ state, its successor is (y_k, x_j), where y_k is the appropriate successor of y_i.
   - If in (y_i, x_j) and y_i is a READ state, reading a, its successor is (y_k, x_l), where
     - y_k is the appropriate successor of y_i on an a
     - $\delta(x_j, a) = x_l$.

7. I's ACCEPT states are those where the y component is ACCEPT *and* the x component is final.

   If the y component is ACCEPT and the x component is not final, the state in I is REJECT (or omitted, implying a crash).

Example

- Let L_1 be the CFL EQUAL of words with an equal number of a's and b's.

  Draw its PDA.

- Let $L_2 = (a + b)^*a$.

  Draw its FA.

- Perform the construction of the intersection PDA.