CS 5000: Lecture 13

Vladimir Kulyukin
Department of Computer Science
Utah State University

Outline

- Applications of the CFL Pumping Lemma
- Closure properties of CFL's
- CNF conversion example from Lecture 11 with more notes.

The CFL Pumping Lemma

Lemma: If L is a context-free language, then there exists some $k \in N$ such that for all $z \in L$ such that $|z| \geq k$ there exists $uvwxy$ such that

1. $z = uvwxy$;
2. $|vwx| \leq k$;
3. v and x are not both empty;
4. $(\forall \mathrm{i} \geq 0)\left[uv^i wx^i y \in A\right]$.

Example

- Show that L = $a^n b^n c^n$ is not context-free.
- Assume that L is CF.
- Let k be the constant of the pumping lemma.
- Consider $z = a^k b^k c^k$. Then $|z| \geq k$.
- By the pumping lemma, $z = uvwxy$ where $|vwx| \leq k$ and v and x are not both empty.
- Consider the possible choices for v and x.

Example

- *v* and *x* cannot contain more than one symbol. In other words, neither *v* nor *x* can contain *a*'s and *b*'s or *b*'s and *c*'s.
- Thus, we have several possible choices for *v* and *x.* Below are a few examples:
  1. *v* and *x* consist of *a*'s;
  2. *v* consists of *a*'s and x consists of *b*'s;
  3. *v* and *x* consist of *b*'s;
  4. *v* consists of *a*'s and *x* consists of *c*'s;
  5. *v* and *x* consist of *c*'s;
  6. *v* consists of *a*'s and *x* consists of *c*'s.

Example

- There are also several choices when *v* or *x* contain both a’s and b’s or b’s and c’s. Note that we cannot have the case when *v* consists of a’s and b’s and *x* consists of b’s and c’s. One or the other, but not both.
- Take *i*=2 and consider *uvvwxxy*.
- In all possible choices for *v* and *x*, the string *uvvwxxy* is not in *L*.
- This gives us a contradiction, because we assumed that the language was CF and showed that it is not.

Closure Properties of CFL's

- Regular languages are closed under
  - union
  - concatenation
  - Kleene closure
  - intersection

Union

If L_1 and L_2 are any context-free languages, $L_1 \cup L_2$, is context-free.

Proof: Let $G_1 = (V_1, \Sigma_1, S_1, P_1)$ and $G_2 = (V_2, \Sigma_2, S_2, P_2)$ such that $L(G_1) = L_1$ and $L(G_2) = L_2$. Assume that V_1 and V_2 are disjoint. This does not lose generality, because the symbols can be renamed. Construct a new grammar $G = (V, \Sigma, \Sigma, S)$ where:

$V = V_1 \cup V_2 \cup \{S\}$;

$\Sigma = \Sigma_1 \cup \Sigma_2$;

$P = P_1 \cup P_2 \cup \{(S \to S_1), (S \to S_2)\}$.

CNF Conversion (Example from Lecture 11)

Suppose we want to convert the following grammar into CNF.

S ➔ bA

S ➔ aB

A ➔ bAA

A ➔ aS

A ➔ a

B ➔ aBB

B ➔ bS

B ➔ b

CNF Conversion: Step 1

- We leave productions of the form V ➔ z, where V is a variable and z is a terminal, alone. They are already in the required form and can be ported into the CNF grammar unchanged. Thus, we port the following two productions unchanged:
- A ➔ a
- B ➔ b

CNF Conversion: Step 2

- In every production that has a terminal in the right-hand side, we replace that terminal with the left-hand side of a new production that generates just that terminal. Thus:
  - Replace S ➔ bA with S ➔ C_bA; C_b ➔ b
  - Replace S ➔ aB with S ➔ C_aB; C_a ➔ a
  - Replace A ➔ bAA with A ➔ C_bAA
  - Replace A ➔ aS with A ➔ C_aS; C_a ➔ a
  - Replace B ➔ aBB with B ➔ C_aBB
  - Replace B ➔ bS with B ➔ C_bS

CNF Conversion: Step 3

- In step 3, we make sure that every production that has only variables on the right-hand side is replaced with a set of productions each of which has only two variables on the right-hand side:
- Replace $A \rightarrow C_bAA$ with
  - $A \rightarrow C_bD_1$ and $D_1 \rightarrow AA$
- Replace $B \rightarrow C_aBB$ with
  - $B \rightarrow C_a D_2$ and $D_2 \rightarrow BB$

CNF Conversion: Step 4

- In step 4, we combine all the productions generated in steps 1, 2, 3 into one CNF grammar:
- $S \rightarrow C_bA \mid C_aB$
- $A \rightarrow C_aS \mid C_bD_1 \mid a$
- $B \rightarrow C_bS \mid C_aD_2 \mid b$
- $D_1 \rightarrow AA$
- $D_2 \rightarrow BB$
- $C_a \rightarrow a$
- $C_b \rightarrow b$

Suggested Reading

- Sections 10.4 and 10.5.