Chapter 17

Context-Free Languages

17.1 Closure Under Unions

We will now prove some properties of CFLs.

Theorem 36 *If L_1 and L_2 are CFLs, then their union $L_1 + L_2$ is a CFL.*

Proof. By grammars.

- L_1 CFL implies that L_1 has a CFG, CFG_1, that generates it.
- Assume that the nonterminals in CFG_1 are $S, A, B, C, \ldots$.
- Change the nonterminals in CFG_1 to $S_1, A_1, B_1, C_1, \ldots$.
- Do not change the terminals in the CFG_1.
- L_2 CFL implies that L_2 has a CFG, CFG_2, that generates it.
- Assume that the nonterminals in CFG_2 are $S, A, B, C, \ldots$.
- Change the nonterminals in CFG_2 to $S_2, A_2, B_2, C_2, \ldots$.
- Do not change the terminals in the CFG_2.
- Now CFG_1 and CFG_2 have nonintersecting sets of nonterminals.
- We create a CFG for $L_1 + L_2$ as follows:

- Include all of the nonterminals $S_1, A_1, B_1, C_1, \ldots$ and $S_2, A_2, B_2, C_2, \ldots$.
- Include all of the productions from CFG_1 and CFG_2.
- Create a new nonterminal S and a production

$$S \rightarrow S_1 \mid S_2$$

- To see that this new CFG generates $L_1 + L_2$,
  - note that any word in language L_i, $i = 1, 2$, can be generated by first using the production $S \rightarrow S_i$
  - also, since there is no overlap in the use of nonterminals in CFG_1 and CFG_2, once we start a derivation with the production $S \rightarrow S_1$, we can only use the productions originally in CFG_1 and cannot use any of the productions from CFG_2, and so we can only produce words in L_1.
  - Similar situation occurs when we start a derivation with the production $S \rightarrow S_2$.

■

Example:
CFG_1 for L_1

$$\begin{aligned} S &\rightarrow SS \mid AaAb \mid BBB \mid \Lambda \\ A &\rightarrow SaS \mid bBb \mid abba \\ B &\rightarrow SSS \mid baab \end{aligned}$$

CFG_2 for L_2

$$\begin{aligned} S &\rightarrow aS \mid aAba \mid BbB \mid \Lambda \\ A &\rightarrow aSa \mid abab \\ B &\rightarrow BabaB \mid bb \end{aligned}$$

To construct CFG for $L_1 + L_2$

- transform CFG_1

$$\begin{aligned} S_1 &\rightarrow S_1S_1 \mid A_1aA_1b \mid B_1B_1B_1 \mid \Lambda \\ A_1 &\rightarrow S_1aS_1 \mid bB_1b \mid abba \\ B_1 &\rightarrow S_1S_1S_1 \mid baab \end{aligned}$$

- transform CFG_2

$$\begin{aligned} S_2 &\rightarrow aS_2 \mid aA_2ba \mid Bb_2B_2 \mid \Lambda \\ A_2 &\rightarrow aS_2a \mid abab \\ B_2 &\rightarrow B_2abaB_2 \mid bb \end{aligned}$$

- construct CFG for $L_1 + L_2$:

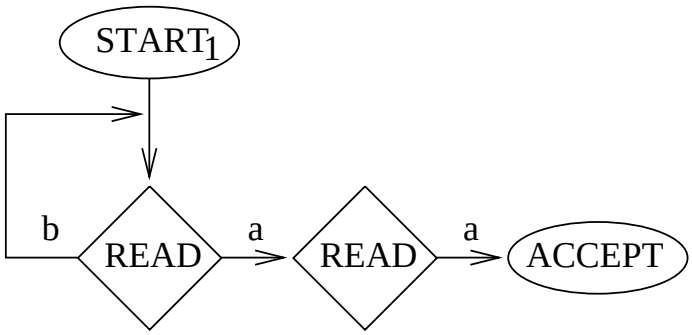
$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow S_1S_1 \mid A_1aA_1b \mid B_1B_1B_1 \mid \Lambda \\ A_1 &\rightarrow S_1aS_1 \mid bB_1b \mid abba \\ B_1 &\rightarrow S_1S_1S_1 \mid baab \\ S_2 &\rightarrow aS_2 \mid aA_2ba \mid Bb_2B_2 \mid \Lambda \\ A_2 &\rightarrow aS_2a \mid abab \\ B_2 &\rightarrow B_2abaB_2 \mid bb \end{aligned}$$

Proof. (of Theorem 36 by machines)

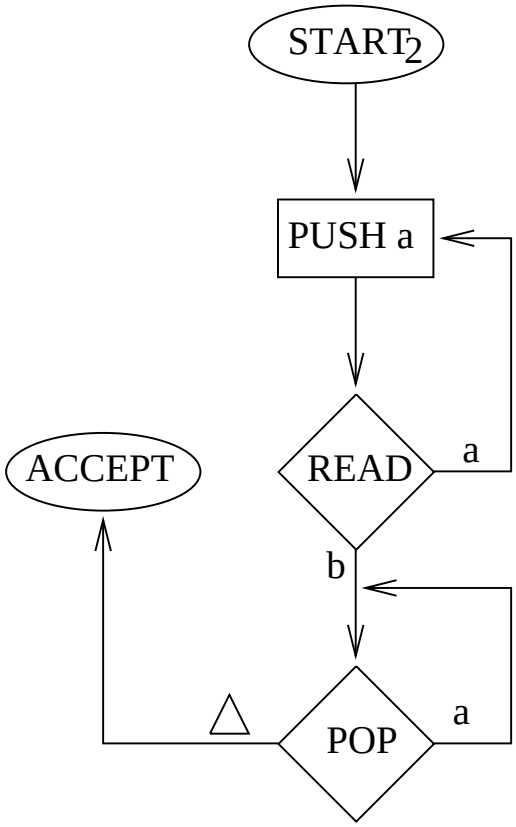
- Since L_1 is CFL, Theorem 30 implies that there exists some PDA, PDA_1, that accepts L_1.
- Since L_2 is CFL, Theorem 30 implies that there exists some PDA, PDA_2, that accepts L_2.
- Construct new PDA_3 to accept $L_1 + L_2$ by combining PDA_1 and PDA_2 into one machine by coalescing START states of PDA_1 and PDA_2 into a single START state.
- Note that once we leave the START state of PDA_3, we can never come back to the START state.
- Also, there is no way to cross over from PDA_1 to PDA_2.
- Hence, any word accepted by PDA_3 must also be accepted by either PDA_1 or PDA_2.
- Also, it is obvious that any word accepted by either PDA_1 or PDA_2 will be accepted by PDA_3.

■

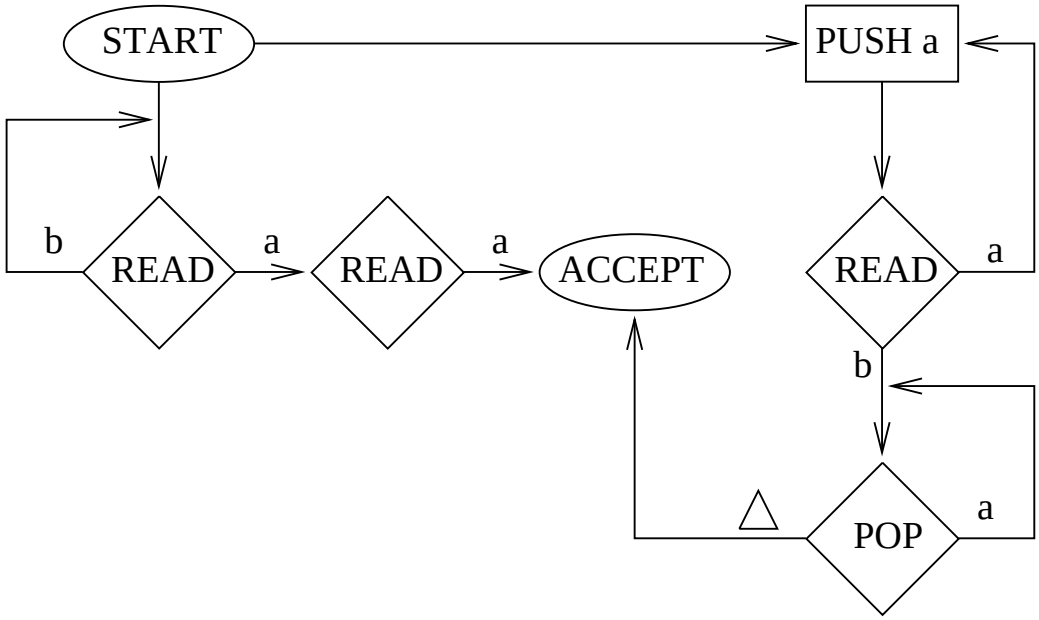
Example:
PDA$_1$ for L_1:



PDA$_2$ for L_2:



PDA$_3$ for $L_1 + L_2$:



17.2 Closure Under Concatenations

Theorem 37 *If L_1 and L_2 are CFLs, then L_1L_2 is a CFL.*

Proof. By grammars.

- L_1 CFL implies that L_1 has a CFG, CFG$_1$, that generates it.
- Assume that the nonterminals in CFG$_1$ are $S, A, B, C, \ldots$.
- Change the nonterminals in CFG$_1$ to $S_1, A_1, B_1, C_1, \ldots$.
- Do not change the terminals in the CFG$_1$.
- L_2 CFL implies that L_2 has a CFG, CFG$_2$, that generates it.
- Assume that the nonterminals in CFG$_2$ are $S, A, B, C, \ldots$.
- Change the nonterminals in CFG$_2$ to $S_2, A_2, B_2, C_2, \ldots$.

- Do not change the terminals in the CFG_2.
- Now CFG_1 and CFG_2 have nonintersecting sets of nonterminals.
- We create a CFG for L_1L_2 as follows:
  - Include all of the nonterminals $S_1, A_1, B_1, C_1, \ldots$ and $S_2, A_2, B_2, C_2, \ldots$.
  - Include all of the productions from CFG_1 and CFG_2.
  - Create a new nonterminal S and a production

$$S \rightarrow S_1S_2$$

- To see that this new CFG generates L_1L_2,
  - Obviously, we can generated any word in L_1L_2 using our new CFG.
  - also, since there is no overlap in the use of nonterminals in CFG_1 and CFG_2, once we start a derivation with the production $S \rightarrow S_1S_2$, the S_1 part will generate a word from L_1 and the S_2 part will generate a word from L_2.
  - hence, any word generated by the new CFG will be in L_1L_2.

■

Example:
CFG_1 for L_1

$$\begin{aligned} S &\rightarrow SS \mid AaAb \mid BBB \mid \Lambda \\ A &\rightarrow SaS \mid bBb \mid abba \\ B &\rightarrow SSS \mid baab \end{aligned}$$

CFG_2 for L_2

$$\begin{aligned} S &\rightarrow aS \mid aAba \mid BbB \mid \Lambda \\ A &\rightarrow aSa \mid abab \\ B &\rightarrow BabaB \mid bb \end{aligned}$$

To construct CFG for L_1L_2

- transform CFG_1

$$\begin{aligned} S_1 &\rightarrow S_1S_1 \mid A_1aA_1b \mid B_1B_1B_1 \mid \Lambda \\ A_1 &\rightarrow S_1aS_1 \mid bB_1b \mid abba \\ B_1 &\rightarrow S_1S_1S_1 \mid baab \end{aligned}$$

- transform CFG_2

$$\begin{aligned} S_2 &\rightarrow aS_2 \mid aA_2ba \mid Bb_2B_2 \mid \Lambda \\ A_2 &\rightarrow aS_2a \mid abab \\ B_2 &\rightarrow B_2abaB_2 \mid bb \end{aligned}$$

- construct CFG for L_1L_2:

$$\begin{aligned} S &\rightarrow S_1S_2 \\ S_1 &\rightarrow S_1S_1 \mid A_1aA_1b \mid B_1B_1B_1 \mid \Lambda \\ A_1 &\rightarrow S_1aS_1 \mid bB_1b \mid abba \\ B_1 &\rightarrow S_1S_1S_1 \mid baab \\ S_2 &\rightarrow aS_2 \mid aA_2ba \mid Bb_2B_2 \mid \Lambda \\ A_2 &\rightarrow aS_2a \mid abab \\ B_2 &\rightarrow B_2abaB_2 \mid bb \end{aligned}$$

Remarks:

- Difficult to prove Theorem 37 by machines.
- Cannot just combine PDA_1 and PDA_2 by removing the ACCEPT state of PDA_1 and replacing it with the START state of PDA_2.
- Problem is we can reach the ACCEPT state of PDA_1 while there are still unread characters on the input TAPE and there are still characters on the STACK.
- Thus, when we go to PDA_2, we may process the last part of the word in L_1 and the entire word in L_2 and incorrectly accept or reject the entire word.

17.3 Closure Under Kleene Star

Theorem 38 *If L is a CFL, then L^* is a CFL.*

Proof.

- Since L is a CFL, by definition there is some CFG that generates L.
- Suppose CFG for L has nonterminals $S, A, B, C, \ldots$.
- Change the nonterminal S to S_1.
- We create a new CFG for L^* as follows:
  - Include all the nonterminals $S_1, A, B, C, \ldots$ from the CFG for L.
  - Include all of the productions from the CFG for L.
  - Add the new nonterminal S and the new production

$$S \rightarrow S_1 S \mid \Lambda$$

- We can repeat last production

$$S \rightarrow S_1 S \rightarrow S_1 S_1 S \rightarrow S_1 S_1 S_1 S \rightarrow S_1 S_1 S_1 S_1 S \rightarrow S_1 S_1 S_1 S_1 \Lambda \rightarrow S_1 S_1 S_1 S_1$$

- Note that any word in L^* can be generated by the new CFG.
- To show that any word generated by the new CFG is in L^*, note that each of the S_1 above generates a word in L.
- Also, there is no interaction between the different S_1's.

■

Example: CFG for L:

$$\begin{aligned} S &\rightarrow AaAb \mid BBB \mid \Lambda \\ A &\rightarrow SaS \mid bBb \mid abba \\ B &\rightarrow SSS \mid baab \end{aligned}$$

Convert CFG for L:

$$\begin{aligned} S_1 &\rightarrow AaAb \mid BBB \mid \Lambda \\ A &\rightarrow S_1aS_1 \mid bBb \mid abba \\ B &\rightarrow S_1S_1S_1 \mid baab \end{aligned}$$

New CFG for L^*:

$$\begin{aligned} S &\rightarrow S_1S \mid \Lambda \\ S_1 &\rightarrow AaAb \mid BBB \mid \Lambda \\ A &\rightarrow S_1aS_1 \mid bBb \mid abba \\ B &\rightarrow S_1S_1S_1 \mid baab \end{aligned}$$

17.4 Intersections

- We now will give an example showing that the intersection of two CFLs may *not* be a CFL.
- To show this, we will need to assume that the language $L_3 = \{a^nb^na^n : n = 0, 1, 2, \ldots\}$ is a non-context-free language. This is shown in the textbook in Chapter 16. L_3 is the set of words with some number of a's, followed by an equal number of b's, and ending with the same number of a's.

Example:

- Let L_1 be generated by the following CFG:

$$\begin{aligned} S &\rightarrow XY \\ X &\rightarrow aXb \mid \Lambda \\ Y &\rightarrow aY \mid \Lambda \end{aligned}$$

  Thus, $L_1 = \{a^nb^na^m : n, m \geq 0\}$, which is the set of words that have a clump of a's, followed by a clump of b's, and ending with another clump of a's, where the number of a's at the beginning is the same as the number of b's in the middle. The number of a's at the end of the word is arbitrary, and does not have to equal the number of a's and b's that come before it.

- Let L_2 be generated by the following CFG:

$$\begin{aligned} S &\rightarrow WZ \\ W &\rightarrow aW \mid \Lambda \\ Z &\rightarrow bZa \mid \Lambda \end{aligned}$$

  Thus, $L_2 = \{a^i b^k a^k : i, k \geq 0\}$, which is the set of words that have a clump of a's, followed by a clump of b's, and ending with another clump of a's, where the number of b's in the middle is the same as the number of a's at the end. The number of a's at the beginning of the word is arbitrary, and does not have to equal the number of b's and a's that come after it.

- Note that $L_1 \cap L_2 = L_3$, where $L_3 = \{a^n b^n a^n : n = 0, 1, 2, \ldots\}$, which is a non-context-free language.

■

- However, sometimes the intersection of two CFLs is a CFL.
- For example, suppose that L_1 and L_2 are regular languages. Then Theorem 21 implies that L_1 and L_2 are CFLs. Also, Theorem 12 implies that $L_1 \cap L_2$ is a regular language, and so $L_1 \cap L_2$ is also a CFL by Theorem 21. Thus, here is an example of 2 CFLs whose intersection is a CFL.
- Thus, in general, we cannot say if the intersection of two CFLs is a CFL.

17.5 Complementation

- If L is a CFL, then L' may or may not be a CFL.
- We first show that the complement of a CFL may be a CFL:
  - If L is regular, then L' is also regular by Theorem 11.
  - Also, Theorem 21 implies that both L and L' are CFLs.
- We now show that the complement of a CFL may not be a CFL by contradiction:

  - Suppose that it is always true that if L is a CFL, then L' is a CFL.
  - Suppose that L_1 and L_2 are CFLs.
  - Then by our assumption, we must have that L_1' and L_2' are CFLs.
  - Theorem 36 implies that $L_1' + L_2'$ is a CFL.
  - Then by our assumption, we must have that $(L_1' + L_2')'$ is a CFL.
  - But we know that $(L_1' + L_2')' = L_1 \cap L_2$ by DeMorgan's Law.
  - However, we previously showed that the intersection of two CFLs is not always a CFL, which contradicts the previous two steps.
  - So our assumption that CFLs are always closed under complementation must not be true.

- Thus, in general, we cannot say if the complement of a CFL is a CFL.