1 Closure Properties

1.1 Regular Operations

Union of CFLs

**Proposition 1.** If \( L_1 \) and \( L_2 \) are context-free languages then \( L_1 \cup L_2 \) is also context-free.

**Proof.** Let \( L_1 \) be language recognized by \( G_1 = (V_1, \Sigma, R_1, S_1) \) and \( L_2 \) the language recognized by \( G_2 = (V_2, \Sigma, R_2, S_2) \). Assume that \( V_1 \cap V_2 = \emptyset \); if this assumption is not true, rename the variables of one of the grammars to make this condition true.

We will construct a grammar \( G = (V, \Sigma, R, S) \) such that \( L(G) = L(G_1) \cup L(G_2) \) as follows.

- \( V = V_1 \cup V_2 \cup \{ S \} \), where \( S \notin V_1 \cup V_2 \) (and \( V_1 \cap V_2 = \emptyset \))
- \( R = R_1 \cup R_2 \cup \{ S \rightarrow S_1|S_2 \} \)

We need to show that \( L(G) = L(G_1) \cup L(G_2) \). Consider \( w \in L(G) \). That means there is a derivation \( S \Rightarrow_G w \). Since the only rules involving \( S \) are \( S \rightarrow S_1 \) and \( S \rightarrow S_2 \), this derivation is either of the form \( S \Rightarrow_G S_1 \Rightarrow_G w \) or \( S \Rightarrow_G S_2 \Rightarrow_G w \). Consider the first case. Since the only rules for variables in \( V_1 \) are those belonging to \( R_1 \) and since \( S \Rightarrow_G w \), we have \( S \Rightarrow_{G_1} w \), and so \( w \in L_1 = L(G_1) \). If the derivation \( S \Rightarrow_G w \) is of the form \( S \Rightarrow_G S_2 \Rightarrow_G w \), then by a similar reasoning we can conclude that \( w \in L(G_2) \). Hence if \( w \in L(G) \) then \( w \in L(G_1) \cup L(G_2) \).

Conversely, consider \( w \in L(G_1) \cup L(G_2) \). Suppose \( w \in L(G_1) \); the case that \( w \in L(G_2) \) is similar and skipped. That means that \( S \Rightarrow_{G_1} w \). Since \( R_1 \subseteq R \), we have \( S \Rightarrow_G w \). Thus, we have \( S \Rightarrow_G S_1 \Rightarrow_G w \) which means that \( w \in L(G) \). This completes the proof.

---

**Concatenation, Kleene Closure**

**Proposition 2.** CFLs are closed under concatenation and Kleene closure

**Proof.** Let \( L_1 \) be language generated by \( G_1 = (V_1, \Sigma, R_1, S_1) \) and \( L_2 \) the language generated by \( G_2 = (V_2, \Sigma, R_2, S_2) \). As before we will assume that \( V_1 \cap V_2 = \emptyset \).

**Concatenation** Let \( G = (V, \Sigma, R, S) \) be such that \( V = V_1 \cup V_2 \cup \{ S \} \) (with \( S \notin V_1 \cup V_2 \)), and \( R = R_1 \cup R_2 \cup \{ S \rightarrow S_1S_2 \} \). We will show that \( L(G) = L(G_1)L(G_2) \). Suppose \( w \in L(G) \). Then there is a leftmost derivation \( S \Rightarrow_{lm}^* w \). The form such a derivation is \( S \Rightarrow G \, S_1S_2 \Rightarrow_{lm}^* w_1S_2 \Rightarrow_{lm}^* w_1w_2 = w \). Thus, \( S_1 \Rightarrow_{lm}^* w_1 \) and \( S_2 \Rightarrow_{lm}^* w_2 \). Since the rules in \( R \) restricted to \( V_1 \) are \( R_1 \) and restricted to \( V_2 \) are \( R_2 \), we can conclude that \( S_1 \Rightarrow_{lm}^* w_1 \) and \( S_2 \Rightarrow_{lm}^* w_2 \). Thus, \( w_1 \in L(G_1) \) and \( w_2 \in L(G_2) \) and therefore, \( w = w_1w_2 \in L(G_1)L(G_2) \). On the other hand, if \( w_1 \in L(G_1) \) and \( w_2 \in L(G_2) \) then we have \( S_1 \Rightarrow_{lm} w_1 \) and \( S_2 \Rightarrow_{lm}^* w_2 \). Take \( w = w_1w_2 \in L(G_1)L(G_2) \). Now since \( R_1 \cup R_2 \subseteq R \), we have \( S \Rightarrow_G w_1 \) and \( S \Rightarrow_G w_2 \). Therefore, we have, \( S \Rightarrow_G w_1S_2 \Rightarrow_G w_1w_2 = w \), and so \( w \in L(G) \).
Kleene Closure Let $G = (V = V_1 \cup \{S\}, \Sigma, \mathcal{R} = R_1 \cup \{S \rightarrow SS \mid \epsilon\}, S)$, where $S \notin V_1$. We will show that $L(G) = (L(G_1))^*$. We will show if $w \in L(G)$ then $w \in (L(G_1))^*$ by induction on the length of the leftmost derivation of $w$. For the base case, consider $w$ such that $S \Rightarrow^G w$. Since $S \rightarrow \epsilon$ is the only rule for $S$ whose right-hand side has terminals, this means that $w = \epsilon$. Further, $\epsilon \in (L(G_1))^*$ which establishes the base case. The induction hypothesis assumes that for all strings $w$, if $S \Rightarrow^G w_{\text{im}}$ in $< n$ steps then $w \in (L(G_1))^*$. Consider $w$ such that $S \Rightarrow^G w_{\text{im}}$ in $n$ steps. Any leftmost derivation has the following form: $S \Rightarrow^G SS_1 \Rightarrow^G w_1S_1 \Rightarrow^G w_1w_2 = w$. Now we have $S \Rightarrow^G w_{\text{im}}$ is $< n$ steps (because $S_1 \Rightarrow^G w$ takes at least one step), and $S_1 \Rightarrow^G w_2$. This means that $w_1 \in (L(G_1))^*$ (by induction hypothesis) and $w_2 \in L(G_1)$ (since the only rules in $R$ for variables in $V_1$ are those belonging to $R_1$). Thus, $w = w_1w_2 \in (L(G_1))^*$. For the converse, suppose $w \in (L(G_1))^*$. By definition, this means that there are $w_1, w_2, \ldots, w_n$ (for $n \geq 0$) such that $w_i \in L(G_1)$ for all $i$. Now if $n = 0$ (i.e., $w = \epsilon$) then we have $S \Rightarrow w$ because $S \rightarrow \epsilon$ is a rule. Otherwise, since $w_i \in L(G_1)$, we have $S_1 \Rightarrow w_i$, for each $i$. Since $R_1 \subseteq R$, $S_1 \Rightarrow w_i$. Hence we have the following derivation

\[ S \Rightarrow SS_1 \Rightarrow SS_1 \Rightarrow \cdots \Rightarrow S(S_1)^n \Rightarrow G S_1^n \Rightarrow^G w_1(S_1)^{n-1} \Rightarrow^G \cdots \Rightarrow^G w_1w_2 \cdots w_n = w \]

\[ \square \]

1.2 Intersection and Complementation

Intersection

**Proposition 3.** CFLs are not closed under intersection

**Proof.**
- $L_1 = \{a^i b^j c^i \mid i, j \geq 0\}$ is a CFL
  - Generated by a grammar with rules $S \rightarrow XY; X \rightarrow aXb|\epsilon; Y \rightarrow cY|\epsilon$.

- $L_2 = \{a^i b^j c^i \mid i, j \geq 0\}$ is a CFL.
  - Generated by a grammar with rules $S \rightarrow XY; X \rightarrow aX|\epsilon; Y \rightarrow bYc|\epsilon$.

- But $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$, which we will see soon, is not a CFL. \[ \square \]

Intersection with Regular Languages

**Proposition 4.** If $L$ is a CFL and $R$ is a regular language then $L \cap R$ is a CFL.

**Proof.** Let $P$ be the PDA that accepts $L$, and let $M$ be the DFA that accepts $R$. A new PDA $P'$ will simulate $P$ and $M$ simultaneously on the same input and accept if both accept. Then $P'$ accepts $L \cap R$. 

\[ 2 \]
• The stack of $P'$ is the stack of $P$
• The state of $P'$ at any time is the pair (state of $P$, state of $M$)
• These determine the transition function of $P'$
• The final states of $P'$ are those in which both the state of $P$ and state of $M$ are accepting.

More formally, let $M = (Q_1, \Sigma, \delta_1, q_1, F_1)$ be a DFA such that $L(M) = R$, and $P = (Q_2, \Sigma, \Gamma, \delta_2, q_2, F_2)$ be a PDA such that $L(P) = L$. Then consider $P' = (Q, \Sigma, \Gamma, \delta, q_0, F)$ such that

- $Q = Q_1 \times Q_2$
- $q_0 = (q_1, q_2)$
- $F = F_1 \times F_2$

$$\delta((p,q), x,a) = \begin{cases} \\
\{((p,q'), b) \mid (q', b) \in \delta_2(q, x,a)\} & \text{when } x = \epsilon \\
\{((p', q'), b) \mid p' = \delta_1(p, x) \text{ and } (q', b) \in \delta_2(q, x,a)\} & \text{when } x \neq \epsilon 
\end{cases}$$

One can show by induction on the number of computation steps, that for any $w \in \Sigma^*$

$$\langle q_0, \epsilon \rangle \xrightarrow{w} P' \langle (p,q), \sigma \rangle \text{ iff } q_1 \xrightarrow{w} M p \text{ and } \langle q_2, \epsilon \rangle \xrightarrow{w} P \langle q, \sigma \rangle$$

The proof of this statement is left as an exercise. Now as a consequence, we have $w \in L(P')$ iff $\langle q_0, \epsilon \rangle \xrightarrow{w} P' \langle (p,q), \sigma \rangle$ such that $(p,q) \in F$ (by definition of PDA acceptance) iff $\langle q_0, \epsilon \rangle \xrightarrow{w} P' \langle (p,q), \sigma \rangle$ such that $p \in F_1$ and $q \in F_2$ (by definition of $F$) iff $q_1 \xrightarrow{w} M p$ and $\langle q_2, \epsilon \rangle \xrightarrow{w} P \langle q, \sigma \rangle$ and $p \in F_1$ and $q \in F_2$ (by the statement to be proved as exercise) iff $w \in L(M)$ and $w \in L(P)$ (by definition of DFA acceptance and PDA acceptance).

Why does this construction not work for intersection of two CFLs?

Complementation

**Proposition 5.** Context-free languages are not closed under complementation.

**Proof.** [Proof 1] Suppose CFLs were closed under complementation. Then for any two CFLs $L_1$, $L_2$, we have

- $\overline{L_1}$ and $\overline{L_2}$ are CFL. Then, since CFLs closed under union, $\overline{L_1} \cup \overline{L_2}$ is CFL. Then, again by hypothesis, $\overline{L_1 \cup L_2}$ is CFL.
- i.e., $L_1 \cap L_2$ is a CFL

i.e., CFLs are closed under intersection. Contradiction!

[Proof 2] $L = \{x \mid x \text{ not of the form } ww\}$ is a CFL.

- $L$ generated by a grammar with rules $X \rightarrow a|b$, $A \rightarrow a|XAX$, $B \rightarrow b|XBX$, $S \rightarrow A|B|AB|BA$

But $L = \{ww \mid w \in \{a,b\}^*\}$ we will see is not a CFL!
Set Difference

**Proposition 6.** If $L_1$ is a CFL and $L_2$ is a CFL then $L_1 \setminus L_2$ is not necessarily a CFL

**Proof.** Because CFLs not closed under complementation, and complementation is a special case of set difference. (How?)

**Proposition 7.** If $L$ is a CFL and $R$ is a regular language then $L \setminus R$ is a CFL

**Proof.** $L \setminus R = L \cap \overline{R}$

### 1.3 Homomorphisms

**Homomorphism**

**Proposition 8.** Context free languages are closed under homomorphisms.

**Proof.** Let $G = (V, \Sigma, R, S)$ be the grammar generating $L$, and let $h : \Sigma^* \rightarrow \Gamma^*$ be a homomorphism. A grammar $G' = (V', \Gamma, R', S')$ for generating $h(L)$:

- Include all variables from $G$ (i.e., $V' \supseteq V$), and let $S' = S$
- Treat terminals in $G$ as variables. i.e., for every $a \in \Sigma$
  - Add a new variable $X_a$ to $V'$
  - In each rule of $G$, if $a$ appears in the RHS, replace it by $X_a$
- For each $X_a$, add the rule $X_a \rightarrow h(a)$

$G'$ generates $h(L)$. (Exercise!)

**Example 9.** Let $G$ have the rules $S \rightarrow 0S0|1S1|\epsilon$.

Consider the homorphism $h : \{0,1\}^* \rightarrow \{a,b\}^*$ given by $h(0) = aba$ and $h(1) = bb$.

Rules of $G'$ s.t. $L(G') = L(L(G))$:

\[
\begin{align*}
S & \rightarrow X_0 S X_0 | X_1 S X_1 | \epsilon \\
X_0 & \rightarrow aba \\
X_1 & \rightarrow bb
\end{align*}
\]
1.4 Inverse Homomorphisms

Inverse Homomorphisms

Recall: For a homomorphism \( h \), \( h^{-1}(L) = \{ w \mid h(w) \in L \} \)

Proposition 10. If \( L \) is a CFL then \( h^{-1}(L) \) is a CFL

Proof Idea

For regular language \( L \): the DFA for \( h^{-1}(L) \) on reading a symbol \( a \), simulated the DFA for \( L \) on \( h(a) \). Can we do the same with PDAs?

- Key idea: store \( h(a) \) in a “buffer” and process symbols from \( h(a) \) one at a time (according to the transition function of the original PDA), and the next input symbol is processed only after the “buffer” has been emptied.
- Where to store this “buffer”? In the state of the new PDA!

Proof. Let \( P = (Q, \Delta, \Gamma, \delta, q_0, F) \) be a PDA such that \( L(P) = L \). Let \( h : \Sigma^* \to \Delta^* \) be a homomorphism such that \( n = \max_{a \in \Sigma} |h(a)| \), i.e., every symbol of \( \Sigma \) is mapped to a string under \( h \) of length at most \( n \). Consider the PDA \( P' = (Q', \Sigma, \Gamma, \delta', q'_0, F') \) where

- \( Q' = Q \times \Delta^{\leq n} \), where \( \Delta^{\leq n} \) is the collection of all strings of length at most \( n \) over \( \Delta \).
- \( q'_0 = (q_0, \epsilon) \)
- \( F' = F \times \{ \epsilon \} \)
- \( \delta' \) is given by

\[
\delta'((q, v), x, a) = \begin{cases} \{(q, h(x)), \epsilon\} & \text{if } v = a = \epsilon \\ \{(p, u, b) \mid (p, b) \in \delta(q, y, a)\} & \text{if } v = yu, x = \epsilon, \text{ and } y \in (\Delta \cup \{ \epsilon \}) \end{cases}
\]

and \( \delta'(\cdot) = \emptyset \) in all other cases.

We can show by induction that for every \( w \in \Sigma^* \)

\[
\langle q'_0, \epsilon \rangle \xrightarrow{w} P' \langle (q, v), \sigma \rangle \text{ iff } \langle q_0, \epsilon \rangle \xrightarrow{w'} P \langle q, \sigma \rangle
\]

where \( h(w) = w'v \). Again this induction proof is left as an exercise. Now, \( w \in L(P') \) iff \( \langle q'_0, \epsilon \rangle \xrightarrow{w} P' \langle (q, \epsilon), \sigma \rangle \) where \( q \in F \) (by definition of PDA acceptance and \( F' \)) iff \( \langle q_0, \epsilon \rangle \xrightarrow{h(w)} F \langle q, \sigma \rangle \) (by exercise) iff \( h(w) \in L(P) \) (by definition of PDA acceptance). Thus, \( L(P') = h^{-1}(L(P)) = h^{-1}(L) \). \( \square \)