CFL closure Proofs

Context-Free Languages - Closure Properties (Complete Proofs)

🧠 Overview of CFL Closure Properties

CFLs are CLOSED under:

- Union (∪)
- Concatenation (·)
- Kleene Star (*)

CFLs are NOT CLOSED under:

- Intersection (∩)
- Complement (¯)

Key Insight: The closure properties that work are those that can be implemented by "choosing" or "sequencing" derivations from component grammars. Operations requiring "coordination" between different parts fail.

Theorem 1: CFL Closure Under Union

Statement: If L_1 and L_2 are context-free, then $L_1 \cup L_2$ is context-free.

💡 Proof Idea:

Create a new grammar that can generate strings from either original grammar by adding a new start symbol with productions that "choose" between the two languages.

Proof:

Step 1: Let $G_1 = (V_1, \Sigma_1, R_1, S_1)$ generate L_1 and $G_2 = (V_2, \Sigma_2, R_2, S_2)$ generate L_2.

Step 2: WLOG, assume $V_1 \cap V_2 = \emptyset$ (rename variables if necessary).

Step 3: Construct $G = (V, \Sigma, R, S)$ where:

- $V = V_1 \cup V_2 \cup \{S\}$ (S is new start symbol)
- $\Sigma = \Sigma_1 \cup \Sigma_2$
- $R = R_1 \cup R_2 \cup \{S \to S_1, S \to S_2\}$

Step 4: Prove $L(G) = L_1 \cup L_2$:

$\subseteq$: If $w \in L(G)$, then $S \Rightarrow^* w$. The derivation starts $S \to S_1$ or $S \to S_2$.

- If $S \to S_1$, then $S_1 \Rightarrow^* w$ using only R_1, so $w \in L_1$
- If $S \to S_2$, then $S_2 \Rightarrow^* w$ using only R_2, so $w \in L_2$
- Therefore $w \in L_1 \cup L_2$

$\supseteq$: If $w \in L_1 \cup L_2$:

- If $w \in L_1$, then $S_1 \Rightarrow^* w$, so $S \to S_1 \Rightarrow^* w$
- If $w \in L_2$, then $S_2 \Rightarrow^* w$, so $S \to S_2 \Rightarrow^* w$
- Therefore $w \in L(G)$

Theorem 2: CFL Closure Under Concatenation

Statement: If L_1 and L_2 are context-free, then $L_1 \cdot L_2$ is context-free.

💡 Proof Idea:

Force every string to be decomposed into two parts: first part generated by G_1, second part by G_2. The new start symbol coordinates this sequencing.

Proof:

Step 1: Let $G_1 = (V_1, \Sigma_1, R_1, S_1)$ generate L_1 and $G_2 = (V_2, \Sigma_2, R_2, S_2)$ generate L_2.

Step 2: WLOG, assume $V_1 \cap V_2 = \emptyset$.

Step 3: Construct $G = (V, \Sigma, R, S)$ where:

- $V = V_1 \cup V_2 \cup \{S\}$ (S is new start symbol)
- $\Sigma = \Sigma_1 \cup \Sigma_2$
- $R = R_1 \cup R_2 \cup \{S \to S_1S_2\}$

Step 4: Prove $L(G) = L_1 \cdot L_2$:

$\subseteq$: If $w \in L(G)$, then $S \Rightarrow^* w$.

- The derivation starts $S \to S_1S_2$
- So $S_1S_2 \Rightarrow^* w$
- This means $w = uv$ where $S_1 \Rightarrow^* u$ and $S_2 \Rightarrow^* v$
- Therefore $u \in L_1$, $v \in L_2$, so $w \in L_1 \cdot L_2$

$\supseteq$: If $w \in L_1 \cdot L_2$, then $w = uv$ where $u \in L_1$, $v \in L_2$.

- $S_1 \Rightarrow^* u$ and $S_2 \Rightarrow^* v$

- So $S \to S_1S_2 \Rightarrow^* uv = w$
- Therefore $w \in L(G)$

Example:

```
L₁ = {aⁿbⁿ | n ≥ 0}, L₂ = {cᵐdᵐ | m ≥ 0}
G₁: S₁ → ε | aS₁b
G₂: S₂ → ε | cS₂d

L₁ · L₂: S → S₁S₂
Result: {aⁿbⁿcᵐdᵐ | n,m ≥ 0}
```

Theorem 3: CFL Closure Under Kleene Star

Statement: If L is a context-free language, then L^* is context-free.

💡 Proof Idea:

L^* means "zero or more concatenations of strings from L." We need a grammar that can generate ε (zero strings) or recursively build longer concatenations.

Proof:

Step 1: Let $G = (V, \Sigma, R, S_1)$ generate L.

Step 2: Construct $G' = (V', \Sigma, R', S)$ where:

- $V' = V \cup \{S\}$ (S is new start symbol)
- $R' = R \cup \{S \to \varepsilon, S \to SS_1\}$

Step 3: Prove $L(G') = L^*$:

⊆: If $w \in L(G')$, then $S \Rightarrow^* w$.

- If derivation uses $S \to \varepsilon$, then $w = \varepsilon \in L^*$
- If derivation uses $S \to SS_1$, then $w = uv$ where $S \Rightarrow^* u$ and $S_1 \Rightarrow^* v$
- By induction: $u \in L^*$ and $v \in L$, so $w = uv \in L^*$

⊇: If $w \in L^*$, then either:

- $w = \varepsilon$: Use $S \to \varepsilon$
- $w = w_1w_2...w_k$ where each $w_i \in L$: Use $S \to SS_1 \Rightarrow^* SS_1w_1 \to SS_1w_1 \Rightarrow^* SS_1w_1w_2...w_{k-1} \to S_1w_1w_2...w_{k-1} \Rightarrow^* w_1w_2...w_k = w$

Alternative Construction (often cleaner):

- $R' = R \cup \{S \to \varepsilon \mid S_1S\}$

This generates strings by: $S \to S_1S \to S_1S_1S \to \ldots \to S_1S_1\ldots S_1\varepsilon$

Example:

```
L = {aⁿbⁿ | n ≥ 1}: S₁ → ab | aS₁b
L*: S → ε | S₁S
Result: L* = {(aⁿbⁿ)ᵏ | n ≥ 1, k ≥ 0}
```

Theorem 4: CFLs are NOT Closed Under Intersection

Statement: There exist context-free languages L_1 and L_2 such that $L_1 \cap L_2$ is not context-free.

💡 Proof Idea:

Find two CFLs whose intersection forces a coordination that CFGs cannot handle - typically requiring equal counts of three or more different symbols.

Proof (Classic Counterexample):

Step 1: Define the languages:

- $L_1 = \{a^ib^jc^j \mid i, j \geq 1\}$
- $L_2 = \{a^ib^ic^j \mid i, j \geq 1\}$

Step 2: Show L_1 and L_2 are context-free:

For L_1:

```
S₁ → AC
A → aA | a
C → bCc | bc
```

For L_2:

```
S₂ → AB
A → aAb | ab
B → cB | c
```

Step 3: Find the intersection: $L_1 \cap L_2 = \{a^ib^jc^j \mid i, j \geq 1\} \cap \{a^ib^ic^j \mid i, j \geq 1\} = \{a^ib^ic^i \mid i \geq 1\}$

Step 4: Prove $L_1 \cap L_2$ is not context-free using the CFL Pumping Lemma:

Let p be the pumping length. Consider $s = a^pb^pc^p \in L_1 \cap L_2$.

By the pumping lemma, $s = uvwxy$ where $|vwx| \leq p$, $|vx| \geq 1$, and $uv^iwx^iy \in L_1 \cap L_2$ for all $i \geq 0$.

Case Analysis:

- **Case 1:** vwx spans only one type of symbol (all a's, all b's, or all c's) Then pumping changes count of only one symbol type, breaking the equality requirement.
- **Case 2:** vwx spans two types of symbols Since $|vwx| \leq p$, it cannot span all three types. Pumping changes at most two symbol counts, but we need all three to remain equal.

In all cases, pumping fails to maintain $a^i b^i c^i$ pattern. Therefore $L_1 \cap L_2$ is not context-free.

Theorem 5: CFLs are NOT Closed Under Complement

Statement: There exists a context-free language L such that $\bar{L}$ is not context-free.

💡 Proof Idea:

Use the non-closure under intersection and De Morgan's laws. If CFLs were closed under complement, they would also be closed under intersection.

Proof (Reduction from Intersection):

Step 1: Assume, for contradiction, that CFLs are closed under complement.

Step 2: Let L_1 and L_2 be any context-free languages.

Step 3: By De Morgan's law: $L_1 \cap L_2 = \bar{L}_1 \cup \bar{L}_2$

Step 4: If CFLs are closed under complement:

- $\bar{L}_1$ would be context-free
- $\bar{L}_2$ would be context-free

Step 5: Since CFLs are closed under union:

- $\bar{L}_1 \cup \bar{L}_2$ would be context-free
- Therefore $L_1 \cap L_2$ would be context-free

Step 6: This contradicts Theorem 4 (CFLs not closed under intersection).

Therefore, CFLs are not closed under complement.

Direct Counterexample:

Alternative Proof: Consider $L = \{a^i b^j c^k \mid i \neq j \text{ or } j \neq k\}$

Step 1: Show L is context-free: $L = \{a^i b^j c^k \mid i \neq j\} \cup \{a^i b^j c^k \mid j \neq k\}$

Each component is CFL (can be shown with appropriate grammars), and CFLs are closed under union.

Step 2: Find the complement: $\bar{L} = \{a^i b^j c^k \mid i = j \text{ and } j = k\} = \{a^i b^i c^i \mid i \geq 0\}$

Step 3: We already proved $\{a^i b^i c^i \mid i \geq 1\}$ is not CFL in Theorem 4. Similarly, $\{a^i b^i c^i \mid i \geq 0\}$ is not CFL.

Therefore $\bar{L}$ is not context-free, proving CFLs are not closed under complement.

Summary Table

Operation	Regular Languages	Context-Free Languages	Decidable Languages
Union (∪)	✅ Closed	✅ Closed	✅ Closed
Intersection (∩)	✅ Closed	❌ Not Closed	✅ Closed
Complement (¯)	✅ Closed	❌ Not Closed	✅ Closed
Concatenation (·)	✅ Closed	✅ Closed	✅ Closed
Kleene Star (*)	✅ Closed	✅ Closed	✅ Closed

Key Insight: As we move up the Chomsky hierarchy, we gain expressive power but lose closure properties. This reflects the fundamental trade-off between computational power and structural constraints in formal language theory.