Show that CFLs are closed under inversion using the Bresolin method

Statement

Theorem: The class of context-free languages is closed under the inversion (reversal) operation.

Formal Statement: For any language A, let $A^R = \{w^R \mid w \in A\}$, where w^R denotes the reversal of string w. If A is context-free, then A^R is also context-free.

Bresolin Method: Two Approaches

Approach 1: Grammar Construction (Direct Method)

Lemma: If L is context-free, then L^R is context-free.

Proof Idea:

- Se L è context-free, allora esiste una CFG G che lo genera
- Mostriamo come trasformare G in una CFG equivalente G^R per L^R
- G^R genera esattamente le stringhe reverse di quelle generate da G

Construction:

Let $G = (V, \Sigma, R, S)$ be a CFG for L. We construct $G^R = (V, \Sigma, R^R, S)$ where:

- Variables V and start symbol S remain the same
- Terminal alphabet Σ remains the same
- Rules R^R are obtained by reversing the right-hand side of each rule in R

Transformation Rule: For each production $A \to \alpha$ in R, add $A \to \alpha^R$ to R^R, where α^R is the reversal of string α.

Examples:

- If $A \to aBc$ is in R, then $A \to cb\alpha$ is in R^R
- If $A \to BC$ is in R, then $A \to CB$ is in R^R
- If $A \to \varepsilon$ is in R, then $A \to \varepsilon$ is in R^R ($\varepsilon^R = \varepsilon$)

Correctness:

1. **$L(G^R) \subseteq (L(G))^R$**: Every derivation in G^R corresponds to a "reversed" derivation in G
2. **$(L(G))^R \subseteq L(G^R)$**: Every reversed string from L(G) can be derived in G^R

Approach 2: PDA Construction (Bresolin Equivalence Method)

Lemma: Se un linguaggio è riconosciuto da un PDA, allora il suo reverse è riconosciuto da un PDA modificato.

Proof Idea:

- Se L è context-free, allora esiste un PDA P che lo riconosce
- Mostriamo come trasformare P in un PDA equivalente P^R per L^R
- P^R simula P "all'indietro" usando proprietà della pila

Construction:

Given PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ for L, construct $P^R = (Q', \Sigma, \Gamma', \delta^R, q_0^R, Z_0^R, F^R)$:

1. **New State Set**: $Q' = Q \cup \{q_0^R, q_f^R\}$ where q_0^R is new start state, q_f^R is new final state
2. **New Stack Alphabet**: $\Gamma' = \Gamma \cup \{Z_0^R, \#\}$ where Z_0^R is new stack symbol, # is bottom marker
3. **Transition Construction**:
   - **Phase 1** (Push input onto stack):
     - $\delta^R(q_0^R, a, Z_0^R) = \{(q_0^R, aZ_0^R)\}$ for all $a \in \Sigma$
     - $\delta^R(q_0^R, a, b) = \{(q_0^R, ab)\}$ for all $a \in \Sigma$, $b \in \Gamma'$
     - $\delta^R(q_0^R, \varepsilon, Z_0^R) = \{(q_0, \#Z_0^R)\}$
   - **Phase 2** (Simulate original PDA with reversed input):
     - For each $(q', \gamma) \in \delta(q, a, X)$: add $(q', \gamma\#) \in \delta^R(q, \varepsilon, a\#)$ if $|\gamma| \geq 1$
     - Special handling for stack operations to maintain reversal semantics
   - **Phase 3** (Accept if original accepts):
     - $\delta^R(q, \varepsilon, \#) = \{(q_f^R, \varepsilon)\}$ for all $q \in F$
4. **Accept States**: $F^R = \{q_f^R\}$

Proof of Correctness (Bresolin Style)

Theorem: Un linguaggio è context-free se e solo se esiste un PDA che lo riconosce.

Applying this to our construction:

1. **Direction 1**: L^R is recognized by $P^R \implies L^R$ is context-free
   - P^R is a valid PDA by construction
   - Therefore L^R is context-free by the fundamental theorem
2. **Direction 2**: P^R correctly recognizes L^R
   - **Soundness**: If P^R accepts w, then $w \in L^R$
   - **Completeness**: If $w \in L^R$, then P^R accepts w

Key Insight: The stack naturally reverses the input, allowing simulation of the original PDA on the reversed string.

Implementation Details

CFG Method Example

Original grammar for L = $\{a^n b^n \mid n \geq 0\}$:

```
S → aSb | ε
```

Reversed grammar for L^R = $\{b^n a^n \mid n \geq 0\}$:

```
S → bSa | ε
```

PDA Method Properties

- **Time Complexity**: Linear transformation
- **Space Complexity**: Constant factor increase in states and stack alphabet
- **Determinism**: Non-deterministic PDA required for the construction

Conclusion

Both methods demonstrate that **CFLs are closed under inversion** following the Bresolin methodology:

1. **Grammar approach**: Direct transformation of production rules
2. **PDA approach**: Stack-based simulation using the fundamental CFG↔PDA equivalence

The construction preserves the context-free property through systematic transformation, proving closure under the reversal operation.

Metodo Bresolin: Utilizzare l'equivalenza fondamentale tra CFG e PDA per dimostrare proprietà di chiusura attraverso costruzioni sistematiche.

I Linguaggi Context-Free NON sono chiusi per intersezione

Enunciato Negativo

Teorema: La classe dei linguaggi context-free **NON** è chiusa sotto l'operazione di intersezione.

Dimostrazione: Per controesempio. Mostreremo due linguaggi context-free L_1 e L_2 tali che $L_1 \cap L_2$ non è context-free.

Controesempio Classico

Definizione dei Linguaggi

$L_1 = \{a^n b^n c^m \mid n,m \geq 0\}$

- Stringhe con ugual numero di a e b seguite da qualsiasi numero di c
- *Esempio*: ε, ab, aabb, abc, aabbc, aaabbbccc, ...

$L_2 = \{a^m b^n c^n \mid n,m \geq 0\}$

- Stringhe con qualsiasi numero di a seguite da ugual numero di b e c
- *Esempio*: ε, bc, aabcc, aaabbcc, abbbccc, ...

Verifica che L_1 e L_2 sono Context-Free

Grammatica per L_1:

```
S₁ → AB
A → aAb | ε
B → cB | ε
```

Grammatica per L_2:

```
S₂ → AB
A → aA | ε
B → bBc | ε
```

Entrambe sono chiaramente grammatiche context-free valide.

Calcolo dell'Intersezione

$L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$

Dimostrazione dell'uguaglianza:

(⊆) Se $w \in L_1 \cap L_2$, allora:

- $w \in L_1 \Longrightarrow w = a^n b^n c^m$ per alcuni $n,m \geq 0$
- $w \in L_2 \Longrightarrow w = a^k b^j c^j$ per alcuni $k,j \geq 0$

Poiché w è la stessa stringa: $a^n b^n c^m = a^k b^j c^j$

- Per uguaglianza di stringhe: n = k, n = j, m = j
- Quindi: n = m = j, cioè $w = a^n b^n c^n$

(⊇) Se $w = a^n b^n c^n$, allora:

- $w \in L_1$ perché $w = a^n b^n c^n$ (con n a's, n b's, n c's)
- $w \in L_2$ perché $w = a^n b^n c^n$ (con n a's, n b's, n c's)
- Quindi $w \in L_1 \cap L_2$

Dimostrazione che $L_1 \cap L_2$ NON è Context-Free

Metodo: Pumping Lemma per linguaggi context-free.

Enunciato del Pumping Lemma: Se L è context-free, allora esiste una costante $p \geq 1$ tale che ogni stringa $s \in L$ con $|s| \geq p$ può essere scritta come s = uvxyz dove:

1. $|vy| > 0$
2. $|vxy| \leq p$
3. $uv^ixy^iz \in L$ per ogni $i \geq 0$

Dimostrazione per assurdo:

1. **Supponiamo** che $L = \{a^n b^n c^n \mid n \geq 0\}$ sia context-free
2. **Applicazione del Pumping Lemma**: Sia p la costante del pumping lemma
3. **Scelta della stringa**: Consideriamo $s = a^p b^p c^p \in L$ con $|s| = 3p \geq p$
4. **Decomposizione**: s = uvxyz con le condizioni del pumping lemma
5. **Analisi dei casi** (dato che $|vxy| \leq p$):

   Caso 1: vxy contiene solo a's

   - Allora v e y contengono solo a's
   - uv^2xy^2z avrebbe più a's che b's e c's $\Longrightarrow uv^2xy^2z \notin L$

   Caso 2: vxy contiene solo b's

   - Allora v e y contengono solo b's
   - uv^2xy^2z avrebbe più b's che a's e c's $\Longrightarrow uv^2xy^2z \notin L$

   Caso 3: vxy contiene solo c's

   - Allora v e y contengono solo c's
   - uv^2xy^2z avrebbe più c's che a's e b's $\Longrightarrow uv^2xy^2z \notin L$

   Caso 4: vxy attraversa due regioni (es. a's e b's)

- Poiché $|vxy| \leq p$, può attraversare al massimo 2 delle 3 regioni
- Se v o y contengono simboli diversi, pompando si altera l'ordine
- Se v e y sono in regioni diverse, pompando si sbilancia il conteggio
- In ogni sottocaso: $uv^2xy^2z \notin L$

6. **Contraddizione**: In tutti i casi, $uv^2xy^2z \notin L$, contraddicendo il pumping lemma
7. **Conclusione**: $L = \{a^nb^nc^n \mid n \geq 0\}$ non è context-free

---

Traduzione in inglese

Statement

Theorem: The class of context-free languages is **NOT** closed under intersection.

Proof: By counterexample. We show two context-free languages L_1 and L_2 such that $L_1 \cap L_2$ is not context-free.

Counterexample

- **$L_1 = \{a^nb^nc^m \mid n,m \geq 0\}$** (context-free)
- **$L_2 = \{a^mb^nc^n \mid n,m \geq 0\}$** (context-free)
- **$L_1 \cap L_2 = \{a^nb^nc^n \mid n \geq 0\}$** (not context-free)

The intersection is proven non-context-free using the pumping lemma for CFLs.

Conseguenze Teoriche

Proprietà di Chiusura dei CFG

Chiusi sotto:

- ✅ Unione ($L_1 \cup L_2$)
- ✅ Concatenazione (L_1L_2)
- ✅ Stella di Kleene (L^*)
- ✅ Omomorfismo
- ✅ Intersezione con linguaggi regolari ($L_1 \cap R$ dove R è regolare)

NON chiusi sotto:

- ❌ **Intersezione** ($L_1 \cap L_2$)
- ❌ **Complemento** ($\bar{L}$)

- ❌ Differenza (L_1 - L_2)

Implicazioni Pratiche

1. **Parsing**: Non possiamo sempre intersecare grammatiche context-free mantenendo la proprietà CF
2. **Linguaggi di programmazione**: Le restrizioni semantiche spesso richiedono controlli context-sensitive
3. **Teoria della computazione**: Separazione netta tra CF e classi superiori

Metodo di Dimostrazione Generale

Per dimostrare **non-chiusura**:

1. Trova due linguaggi CF la cui operazione produce un linguaggio non-CF
2. Usa pumping lemma o altre tecniche per provare la non-CF del risultato
3. Questo costituisce un controesempio sufficiente

La non-chiusura per intersezione è uno dei risultati fondamentali nella teoria dei linguaggi formali e dimostra i limiti intrinseci dei linguaggi context-free.