Automi e Linguaggi Formali

Esercizi su Grammatiche Context-Free e Forme Normali

Gabriel Rovesti
Corso di Laurea in Informatica - Università degli Studi di Padova

Aprile 2025

1 Costruzione di Grammatiche Context-Free

Esercizio 1. Progettare grammatiche context-free per i seguenti linguaggi:

(a) $L_1 = \{a^i b^j c^k \mid i + j = k, i, j, k \geq 1\}$

(b) $L_2 = \{a^n b^m c^p \mid n = m \text{ oppure } m = p, n, m, p \geq 1\}$

(c) $L_3 = \{a^n b^m c^k \mid m = n + k, n, m, k \geq 1\}$

(d) $L_4 = \{w \in \{a, b\}^* \mid \text{ogni prefisso di } w \text{ ha più } a \text{ che } b\}$

Soluzione. **(a)** Per $L_1 = \{a^i b^j c^k \mid i + j = k, i, j, k \geq 1\}$:

$$
\begin{aligned}
S &\to aSc \mid T \\
T &\to bTc \mid bc
\end{aligned}
$$

Questa grammatica funziona così:

- $S \to aSc$ genera coppie di a e c
- $S \to T$ passa alla generazione di coppie di b e c
- $T \to bTc$ genera coppie di b e c
- $T \to bc$ assicura che ci sia almeno un b e un c

Complessivamente, la grammatica genera stringhe dove il numero di c è uguale alla somma del numero di a e b.

(b) Per $L_2 = \{a^n b^m c^p \mid n = m \text{ oppure } m = p, n, m, p \geq 1\}$:

$$
\begin{aligned}
S &\to A \mid B \\
A &\to aAb \mid abC \\
B &\to B_1 c \mid D \\
B_1 &\to aB_1 \mid a \\
C &\to cC \mid c \\
D &\to bDc \mid bc
\end{aligned}
$$

Questa grammatica ha due parti principali:

- La produzione A genera stringhe dove $n = m$, seguite da un numero qualsiasi di c
- La produzione B genera stringhe dove $m = p$, precedute da un numero qualsiasi di a

(c) Per $L_3 = \{a^n b^m c^k \mid m = n + k, n, m, k \geq 1\}$:

$$S \to aSc \mid aT$$
$$T \to aT \mid aUb$$
$$U \to Ub \mid b$$

Il funzionamento:

- $S \to aSc$ genera $a^i b^j c^i$ con pari a e c
- $S \to aT$ permette di aggiungere a in eccesso
- $T \to aT$ aggiunge ulteriori a
- $T \to aUb$ e $U \to Ub \mid b$ generano tanti b quanti sono gli a e c insieme

(d) Per $L_4 = \{w \in \{a, b\}^* \mid \text{ogni prefisso di } w \text{ ha più } a \text{ che } b\}$:

$$S \to aS \mid aSbS \mid \varepsilon$$

Questa grammatica genera stringhe dove ogni prefisso ha più a che b come segue:

- $S \to aS$ aggiunge un a (mantenendo la proprietà)
- $S \to aSbS$ aggiunge un b solo se preceduto da un a in eccesso
- $S \to \varepsilon$ termina la generazione

La grammatica costruisce stringhe che mantengono sempre un "credito" di a non accoppiati, garantendo che ogni prefisso abbia più a che b.

Esercizio 2. Dato il seguente problema reale: progettare una grammatica context-free che generi espressioni aritmetiche ben formate con le seguenti caratteristiche:

- Operatori: +, -, *, / (binari)
- Operandi: numeri interi positivi e variabili (identificatori di una lettera)
- Precedenza standard degli operatori (* e / prima di + e -)
- Associatività a sinistra degli operatori
- Supporto per parentesi tonde che possono modificare la precedenza

Soluzione. Una grammatica context-free per le espressioni aritmetiche ben formate:

$$\begin{aligned}
\langle\text{expr}\rangle &\to \langle\text{expr}\rangle + \langle\text{term}\rangle \mid \langle\text{expr}\rangle - \langle\text{term}\rangle \mid \langle\text{term}\rangle \\
\langle\text{term}\rangle &\to \langle\text{term}\rangle * \langle\text{factor}\rangle \mid \langle\text{term}\rangle / \langle\text{factor}\rangle \mid \langle\text{factor}\rangle \\
\langle\text{factor}\rangle &\to (\langle\text{expr}\rangle) \mid \langle\text{number}\rangle \mid \langle\text{variable}\rangle \\
\langle\text{number}\rangle &\to \langle\text{digit}\rangle \mid \langle\text{number}\rangle\langle\text{digit}\rangle \\
\langle\text{digit}\rangle &\to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
\langle\text{variable}\rangle &\to a \mid b \mid c \mid \ldots \mid z \mid A \mid B \mid C \mid \ldots \mid Z
\end{aligned}$$

Questa grammatica modella:

- Precedenza degli operatori: * e / hanno precedenza maggiore di + e - grazie alla struttura a livelli della grammatica
- Associatività a sinistra: per esempio, $a + b + c$ viene interpretato come $(a + b) + c$ grazie alla ricorsione a sinistra nelle regole ⟨expr⟩ e ⟨term⟩
- Parentesi: la regola ⟨factor⟩ → (⟨expr⟩) permette di sovrascrivere la precedenza standard
- Numeri: possono essere composti da una o più cifre
- Variabili: sono rappresentate da una singola lettera, maiuscola o minuscola

Nota: la grammatica utilizza ricorsione a sinistra, che può essere problematica per alcuni algoritmi di parsing come quello ricorsivo discendente, ma è corretta per la definizione formale ed è utile per implementare l'associatività a sinistra.

2 Determinazione del Linguaggio Generato

Esercizio 3. Per ciascuna delle seguenti grammatiche, determinare il linguaggio generato e dimostrarlo formalmente:

(a) G_1:

$$S \to aSb \mid \varepsilon$$

(b) G_2:

$$S \to aS \mid bS \mid \varepsilon$$

(c) G_3:

$$\begin{aligned} S &\to aAS \mid \varepsilon \\ A &\to SbA \mid a \end{aligned}$$

Soluzione. **(a)** Per la grammatica G_1:

$$S \to aSb \mid \varepsilon$$

Claim: $L(G_1) = \{a^n b^n \mid n \geq 0\}$
Dimostrazione:

- ($L(G_1) \subseteq \{a^n b^n \mid n \geq 0\}$): Dimostriamo per induzione sulla lunghezza della derivazione.

  Base: La derivazione più breve è $S \Rightarrow \varepsilon$, producendo la stringa vuota, che è $a^0 b^0 \in \{a^n b^n \mid n \geq 0\}$.

  Passo induttivo: Supponiamo che ogni derivazione di lunghezza k produca una stringa nella forma $a^n b^n$ per qualche $n \geq 0$. Consideriamo una derivazione di lunghezza $k + 1$. Deve iniziare con $S \Rightarrow aSb$, seguito da una derivazione di S di lunghezza $k - 1$. Per ipotesi induttiva, questa sottoderivazione produce $a^m b^m$ per qualche $m \geq 0$. Quindi, la derivazione completa produce $aa^m b^m b = a^{m+1} b^{m+1}$, che è in $\{a^n b^n \mid n \geq 0\}$.

- $(\{a^n b^n \mid n \geq 0\} \subseteq L(G_1))$: Dimostriamo che $a^n b^n \in L(G_1)$ per ogni $n \geq 0$.

  Base: $n = 0$. Abbiamo $a^0 b^0 = \varepsilon$, e $S \Rightarrow \varepsilon$.

  Passo induttivo: Supponiamo che $a^k b^k \in L(G_1)$ per qualche $k \geq 0$. Allora esiste una derivazione $S \Rightarrow^* a^k b^k$. Consideriamo $a^{k+1} b^{k+1}$. Possiamo ottenere questa stringa con la derivazione $S \Rightarrow aSb \Rightarrow^* aa^k b^k b = a^{k+1} b^{k+1}$. Quindi, $a^{k+1} b^{k+1} \in L(G_1)$.

Quindi, $L(G_1) = \{a^n b^n \mid n \geq 0\}$.
(b) Per la grammatica G_2:

$$S \to aS \mid bS \mid \varepsilon$$

Claim: $L(G_2) = \{a, b\}^*$ (tutte le stringhe sull'alfabeto $\{a, b\}$)
Dimostrazione:

- $(L(G_2) \subseteq \{a, b\}^*)$: Ogni stringa prodotta da G_2 è composta solo dai simboli a e b, quindi $L(G_2) \subseteq \{a, b\}^*$.

- $(\{a, b\}^* \subseteq L(G_2))$: Dimostriamo per induzione sulla lunghezza della stringa.

  Base: La stringa vuota ε è in $L(G_2)$ perché $S \Rightarrow \varepsilon$.

  Passo induttivo: Supponiamo che ogni stringa in $\{a, b\}^*$ di lunghezza k sia in $L(G_2)$. Consideriamo una stringa w di lunghezza $k + 1$. w deve essere nella forma aw' o bw', dove w' è una stringa di lunghezza k. Per ipotesi induttiva, $w' \in L(G_2)$, quindi esiste una derivazione $S \Rightarrow^* w'$. Se $w = aw'$, possiamo derivare w come $S \Rightarrow aS \Rightarrow^* aw'$. Se $w = bw'$, possiamo derivare w come $S \Rightarrow bS \Rightarrow^* bw'$. In entrambi i casi, $w \in L(G_2)$.

Quindi, $L(G_2) = \{a, b\}^*$.
(c) Per la grammatica G_3:

$$S \to aAS \mid \varepsilon$$
$$A \to SbA \mid a$$

Questa grammatica è più complessa. Osserviamo che:

- S può produrre stringhe nella forma $aA\varepsilon$, $aAaA\varepsilon$, $aAaAaA\varepsilon$, ecc.

- A può produrre stringhe nella forma a, SbA, $SbSbA$, ecc.

Analizziamo alcune derivazioni:

$$S \Rightarrow \varepsilon$$
$$S \Rightarrow aAS \Rightarrow aA\varepsilon \Rightarrow aa\varepsilon = aa$$
$$S \Rightarrow aAS \Rightarrow aA(aAS) \Rightarrow aA(aA\varepsilon) \Rightarrow aaa\varepsilon = aaa$$
$$S \Rightarrow aAS \Rightarrow a(SbA)S \Rightarrow a(\varepsilon bA)S \Rightarrow abAS \Rightarrow abaS \Rightarrow aba\varepsilon = aba$$

Dopo un'analisi approfondita, possiamo determinare che $L(G_3)$ è l'insieme di tutte le stringhe che consistono in sequenze di a e b dove ogni prefisso ha un numero di a maggiore o uguale al numero di b, e terminano con un carattere a.

Una dimostrazione formale richiederebbe un'induzione più complessa che esamina la struttura delle derivazioni in G_3.

3 Forma Normale di Chomsky

Esercizio 4. Convertire le seguenti grammatiche in Forma Normale di Chomsky, mostrando tutti i passaggi della trasformazione:

(a) G_1:

$$\begin{aligned} S &\to aaBb \mid \varepsilon \\ B &\to abbS \mid a \end{aligned}$$

(b) G_2:

$$\begin{aligned} S &\to AB \mid C \\ A &\to aA \mid a \\ B &\to Bb \mid b \\ C &\to aCa \mid bCb \mid \varepsilon \end{aligned}$$

Soluzione. **(a)** Grammatica G_1:

$$\begin{aligned} S &\to aaBb \mid \varepsilon \\ B &\to abbS \mid a \end{aligned}$$

Passo 1: Aggiungere un nuovo simbolo iniziale S_0:

$$\begin{aligned} S_0 &\to S \\ S &\to aaBb \mid \varepsilon \\ B &\to abbS \mid a \end{aligned}$$

Passo 2: Eliminare le ε-produzioni: Poiché $S \to \varepsilon$, dobbiamo sostituire S con ε in ogni produzione dove appare:

$$\begin{aligned} S_0 &\to S \mid \varepsilon \\ S &\to aaBb \\ B &\to abbS \mid abb \mid a \end{aligned}$$

Passo 3: Eliminare le produzioni unitarie:

$$\begin{aligned} S_0 &\to S \mid \varepsilon \\ S &\to aaBb \\ B &\to abbS \mid abb \mid a \end{aligned}$$

La produzione unitaria $S_0 \to S$ viene sostituita con le produzioni di S:

$$\begin{aligned} S_0 &\to aaBb \mid \varepsilon \\ S &\to aaBb \\ B &\to abbS \mid abb \mid a \end{aligned}$$

Passo 4: Convertire le produzioni in forma corretta: Introduciamo nuove variabili per i terminali:

$$
\begin{aligned}
A_a &\to a \\
A_b &\to b
\end{aligned}
$$

E nuove variabili per le produzioni lunghe:

$$
\begin{aligned}
S_0 &\to X_1 \mid \varepsilon \\
S &\to X_1 \\
X_1 &\to A_a X_2 \\
X_2 &\to A_a X_3 \\
X_3 &\to B X_4 \\
X_4 &\to A_b \\
B &\to X_5 \mid A_a \\
X_5 &\to A_a X_6 \\
X_6 &\to A_b X_7 \\
X_7 &\to A_b X_8 \\
X_8 &\to S \\
A_a &\to a \\
A_b &\to b
\end{aligned}
$$

Possiamo semplificare un po' sostituendo $X_8 \to S$ con $X_8 \to X_1$. La grammatica finale in Forma Normale di Chomsky è:

$$
\begin{aligned}
S_0 &\to X_1 \mid \varepsilon \\
S &\to X_1 \\
X_1 &\to A_a X_2 \\
X_2 &\to A_a X_3 \\
X_3 &\to B X_4 \\
X_4 &\to A_b \\
B &\to X_5 \mid A_a \\
X_5 &\to A_a X_6 \\
X_6 &\to A_b X_7 \\
X_7 &\to A_b X_1 \\
A_a &\to a \\
A_b &\to b
\end{aligned}
$$

(b) Grammatica G_2:

$$
\begin{aligned}
S &\to AB \mid C \\
A &\to aA \mid a \\
B &\to Bb \mid b \\
C &\to aCa \mid bCb \mid \varepsilon
\end{aligned}
$$

Passo 1: Aggiungere un nuovo simbolo iniziale:

$$\begin{aligned}
S_0 &\to S \\
S &\to AB \mid C \\
A &\to aA \mid a \\
B &\to Bb \mid b \\
C &\to aCa \mid bCb \mid \varepsilon
\end{aligned}$$

Passo 2: Eliminare le ε-produzioni: Da $C \to \varepsilon$, sostituiamo C con ε in $S \to C$:

$$\begin{aligned}
S_0 &\to S \\
S &\to AB \mid C \mid \varepsilon \\
A &\to aA \mid a \\
B &\to Bb \mid b \\
C &\to aCa \mid bCb
\end{aligned}$$

Da $S \to \varepsilon$, sostituiamo S con ε in $S_0 \to S$:

$$\begin{aligned}
S_0 &\to S \mid \varepsilon \\
S &\to AB \mid C \mid \varepsilon \\
A &\to aA \mid a \\
B &\to Bb \mid b \\
C &\to aCa \mid bCb \mid aa \mid bb
\end{aligned}$$

Passo 3: Eliminare le produzioni unitarie: Da $S_0 \to S$, sostituendo con le produzioni di S:

$$\begin{aligned}
S_0 &\to AB \mid C \mid \varepsilon \\
S &\to AB \mid C \mid \varepsilon \\
A &\to aA \mid a \\
B &\to Bb \mid b \\
C &\to aCa \mid bCb \mid aa \mid bb
\end{aligned}$$

Da $S \to C$, sostituendo con le produzioni di C:

$$\begin{aligned}
S_0 &\to AB \mid aCa \mid bCb \mid aa \mid bb \mid \varepsilon \\
S &\to AB \mid aCa \mid bCb \mid aa \mid bb \mid \varepsilon \\
A &\to aA \mid a \\
B &\to Bb \mid b \\
C &\to aCa \mid bCb \mid aa \mid bb
\end{aligned}$$

Passo 4: Convertire le produzioni in forma corretta: Introduciamo nuove variabili per i terminali:

$$\begin{aligned}
A_a &\to a \\
A_b &\to b
\end{aligned}$$

E nuove variabili per le produzioni lunghe:

$$
\begin{aligned}
S_0 &\to AB \mid X_1 \mid X_2 \mid X_3 \mid X_4 \mid \varepsilon \\
S &\to AB \mid X_1 \mid X_2 \mid X_3 \mid X_4 \mid \varepsilon \\
X_1 &\to A_a X_5 \\
X_5 &\to C A_a \\
X_2 &\to A_b X_6 \\
X_6 &\to C A_b \\
X_3 &\to A_a A_a \\
X_4 &\to A_b A_b \\
A &\to A_a A \mid A_a \\
B &\to B A_b \mid A_b \\
C &\to X_1 \mid X_2 \mid X_3 \mid X_4 \\
A_a &\to a \\
A_b &\to b
\end{aligned}
$$

La grammatica finale in Forma Normale di Chomsky è complessa ma correttamente formata, con ciascuna produzione che contiene o esattamente due variabili o un singolo terminale.

Esercizio 5. Dato il seguente linguaggio $L = \{a^n b^m c^n \mid m, n \geq 1\}$:

(a) Costruire una grammatica context-free per L.

(b) Trasformare la grammatica in Forma Normale di Chomsky.

(c) Mostrare come la grammatica in Forma Normale di Chomsky deriva la stringa $aabcc$.

Soluzione. **(a)** Una grammatica context-free per $L = \{a^n b^m c^n \mid m, n \geq 1\}$:

$$
\begin{aligned}
S &\to aSc \mid B \\
B &\to bB \mid b
\end{aligned}
$$

Questa grammatica genera $a^n b^m c^n$ come segue:

- $S \to aSc$ genera coppie di a e c nella forma $a^n S c^n$
- $S \to B$ termina la generazione di a e c e passa a generare una sequenza di b
- $B \to bB \mid b$ genera una o più b

(b) Trasformazione in Forma Normale di Chomsky:
Passo 1: Introduciamo un nuovo simbolo iniziale:

$$
\begin{aligned}
S_0 &\to S \\
S &\to aSc \mid B \\
B &\to bB \mid b
\end{aligned}
$$

Passo 2: Non ci sono ε-produzioni da eliminare.

Passo 3: Eliminiamo le produzioni unitarie: Da $S \to B$, sostituiamo con le produzioni di B:

$$\begin{aligned}
S_0 &\to S \\
S &\to aSc \mid bB \mid b \\
B &\to bB \mid b
\end{aligned}$$

Da $S_0 \to S$, sostituiamo con le produzioni di S:

$$\begin{aligned}
S_0 &\to aSc \mid bB \mid b \\
S &\to aSc \mid bB \mid b \\
B &\to bB \mid b
\end{aligned}$$

Passo 4: Convertiamo le produzioni nella forma corretta:

$$\begin{aligned}
A_a &\to a \\
A_b &\to b \\
A_c &\to c
\end{aligned}$$

Introduciamo nuove variabili per le produzioni lunghe:

$$\begin{aligned}
S_0 &\to X_1 \mid X_2 \mid A_b \\
S &\to X_1 \mid X_2 \mid A_b \\
X_1 &\to A_a X_3 \\
X_3 &\to S A_c \\
X_2 &\to A_b B \\
B &\to X_2 \mid A_b \\
A_a &\to a \\
A_b &\to b \\
A_c &\to c
\end{aligned}$$

La grammatica finale in Forma Normale di Chomsky è:

$$\begin{aligned}
S_0 &\to X_1 \mid X_2 \mid A_b \\
S &\to X_1 \mid X_2 \mid A_b \\
X_1 &\to A_a X_3 \\
X_3 &\to S A_c \\
X_2 &\to A_b B \\
B &\to X_2 \mid A_b \\
A_a &\to a \\
A_b &\to b \\
A_c &\to c
\end{aligned}$$

(c) Derivazione della stringa *aabcc* usando la grammatica in Forma Normale di Chomsky:

$$\begin{aligned} S_0 &\Rightarrow X_1 \\ &\Rightarrow A_a X_3 \\ &\Rightarrow a X_3 \\ &\Rightarrow a S A_c \\ &\Rightarrow a X_1 A_c \\ &\Rightarrow a A_a X_3 A_c \\ &\Rightarrow aa X_3 A_c \\ &\Rightarrow aa S A_c A_c \\ &\Rightarrow aa A_b A_c A_c \\ &\Rightarrow aab A_c A_c \\ &\Rightarrow aabc A_c \\ &\Rightarrow aabcc \end{aligned}$$

Questa è una possibile derivazione di $aabcc$ nella grammatica in Forma Normale di Chomsky. Osserviamo che ogni passo della derivazione applica una singola produzione, generando esattamente un terminale o sostituendo una variabile con due variabili, come richiesto dalla definizione della Forma Normale di Chomsky.

4 Dimostrazione di Correttezza

Esercizio 6. Dimostrare che il linguaggio $L = \{a^n b^n c^m \mid n, m \geq 1\}$ è context-free costruendo una grammatica context-free e dimostrando formalmente che essa genera esattamente L.

Soluzione. Proponiamo la seguente grammatica G per il linguaggio $L = \{a^n b^n c^m \mid n, m \geq 1\}$:

$$\begin{aligned} S &\to AB \\ A &\to aAb \mid ab \\ B &\to cB \mid c \end{aligned}$$

Dimostreremo formalmente che $L(G) = L$, ovvero $L(G) = \{a^n b^n c^m \mid n, m \geq 1\}$.
Parte 1: $L(G) \subseteq \{a^n b^n c^m \mid n, m \geq 1\}$
Dimostriamo per induzione strutturale sulle derivazioni in G che:

- Per ogni derivazione $A \Rightarrow^* w$, w è della forma $a^n b^n$ con $n \geq 1$
- Per ogni derivazione $B \Rightarrow^* w$, w è della forma c^m con $m \geq 1$
- Per ogni derivazione $S \Rightarrow^* w$, w è della forma $a^n b^n c^m$ con $n, m \geq 1$

Per A:

- Base: $A \Rightarrow ab$. Chiaramente, $ab = a^1 b^1$ è della forma richiesta.
- Passo induttivo: Supponiamo che $A \Rightarrow^* a^k b^k$ per qualche $k \geq 1$. Consideriamo la derivazione $A \Rightarrow aAb \Rightarrow^* aa^k b^k b = a^{k+1} b^{k+1}$. Questa è ancora della forma $a^n b^n$ con $n \geq 1$.

Per B:

- Base: $B \Rightarrow c$. Chiaramente, $c = c^1$ è della forma richiesta.
- Passo induttivo: Supponiamo che $B \Rightarrow^* c^k$ per qualche $k \geq 1$. Consideriamo la derivazione $B \Rightarrow cB \Rightarrow^* cc^k = c^{k+1}$. Questa è ancora della forma c^m con $m \geq 1$.

Per S: $S \Rightarrow AB \Rightarrow^* a^n b^n c^m$ per qualche $n, m \geq 1$, per quanto dimostrato sopra.

Quindi, ogni stringa in $L(G)$ è della forma $a^n b^n c^m$ con $n, m \geq 1$, ovvero $L(G) \subseteq \{a^n b^n c^m \mid n, m \geq 1\}$.

Parte 2: $\{a^n b^n c^m \mid n, m \geq 1\} \subseteq L(G)$

Dimostriamo che per ogni $n, m \geq 1$, $a^n b^n c^m \in L(G)$.

Sappiamo già che:

- Per ogni $n \geq 1$, $A \Rightarrow^* a^n b^n$
- Per ogni $m \geq 1$, $B \Rightarrow^* c^m$

Quindi, per ogni $n, m \geq 1$:

$$\begin{aligned} S &\Rightarrow AB \\ &\Rightarrow^* a^n b^n B \\ &\Rightarrow^* a^n b^n c^m \end{aligned}$$

Così, $a^n b^n c^m \in L(G)$ per ogni $n, m \geq 1$, ovvero $\{a^n b^n c^m \mid n, m \geq 1\} \subseteq L(G)$.

Combinando le due parti, abbiamo dimostrato che $L(G) = \{a^n b^n c^m \mid n, m \geq 1\} = L$.

Esercizio 7. Sia L_1 un linguaggio context-free e L_2 un linguaggio regolare. Dimostrare che $L_1 \cap L_2$ è context-free costruendo una grammatica context-free per un esempio specifico dove $L_1 = \{a^n b^n \mid n \geq 0\}$ e $L_2 = \{a^i b^j \mid i \leq 3, j \geq 0\}$.

Soluzione. Dobbiamo dimostrare che $L_1 \cap L_2$ è context-free, dove $L_1 = \{a^n b^n \mid n \geq 0\}$ e $L_2 = \{a^i b^j \mid i \leq 3, j \geq 0\}$.

Il linguaggio $L_1 \cap L_2 = \{a^n b^n \mid 0 \leq n \leq 3\}$ è l'insieme di stringhe che hanno lo stesso numero di a e b, con al massimo 3 a.

Possiamo costruire una grammatica G che genera esattamente $L_1 \cap L_2$:

$$\begin{aligned} &S \to \varepsilon \mid aAb \mid aaBb \mid aaaCb \\ &A \to \varepsilon \\ &B \to \varepsilon \\ &C \to \varepsilon \end{aligned}$$

Questa grammatica genera le stringhe:

- ε (quando $n = 0$)
- ab (quando $n = 1$)
- $aabb$ (quando $n = 2$)
- $aaabbb$ (quando $n = 3$)

Per dimostrare formalmente che $L(G) = L_1 \cap L_2$, dobbiamo verificare entrambe le inclusioni:

1. $L(G) \subseteq L_1 \cap L_2$: Ogni stringa generata da G è nella forma $a^n b^n$ con $0 \leq n \leq 3$, quindi appartiene sia a L_1 che a L_2.

2. $L_1 \cap L_2 \subseteq L(G)$: Ogni stringa in $L_1 \cap L_2$ deve essere nella forma $a^n b^n$ con $0 \leq n \leq 3$. La grammatica G genera esattamente queste stringhe, quindi ogni stringa in $L_1 \cap L_2$ è in $L(G)$.

Quindi, $L(G) = L_1 \cap L_2$, dimostrando che $L_1 \cap L_2$ è context-free.

Nota: Questo esempio specifico è semplice perché l'intersezione $L_1 \cap L_2$ è un linguaggio finito. In generale, l'intersezione di un linguaggio context-free e un linguaggio regolare è sempre context-free, ma la dimostrazione generale richiede l'uso di automi a pila e automi a stati finiti.