Linguaggi di Programmazione I: Compilatori

Esercizi svolti - Grammatiche libere dal contesto

Esercizio 1 *Si dimostri che il linguaggio* $L_1 = \{w \in \{a,b\}^* \mid \mid w \mid_a = \mid w \mid_b\}$ *non è regolare, ma è libero dal contesto.*

Svolgimento 1 *Per dimostrare la non appartenenza di* L_1 *all'insieme dei linguaggi regolari, applichiamo il pumping lemma, di cui riportiamo in primo luogo l'enunciato.*

Se L *è un linguaggio regolare, allora esiste una costante* p *tale che per* $z \in L$, $\mid z \mid \geq p$ $\exists u, v, w \in T^*$ *tali che:*

1. $z = uvw$
2. $\mid uv \mid \leq p$
3. $uv^i w \in L$, $i \geq 0$

Mostreremo ora che per il linguaggio L_1 *non può esistere una tale* p*. Fissata* $p > 1$ *qualsiasi, consideriamo la stringa* $a^p b^p \in L_1$*, di lunghezza ovviamente maggiore di* p*, e mostriamo che non esiste nessuna scomposizione di tale stringa che soddisfi le condzioni del pumping lemma.*

Infatti, per la condizione 2, deve essere $u = a^{p-(k+r)}$, $v = a^k$, *e* $w = a^r b^p$. *Ora, per qualunque* $i \geq 0$, *si dovrebbe avere* $a^{p-(k+r)} a^{ik} a^r b^p = a^{p+k(i-1)} b^p \in L$. *Ma questo non è chiaramente vera se* $i \neq 1$.

Resta ora da mostrare che L_1 *sia libero dal contesto. A questo scopo consideriamo la grammatica* G *definita dalle regole* $S \to aSbS$, $S \to bSaS$, $S \to \lambda$ *e dimostriamo che* $L(G) = L_1$*. In primo luogo mostriamo che* $L(G) \subseteq L_1$*, cioè che ogni parola generata dalla grammatica appartiene al linguaggio. Infatti osserviamo che ogni regola introduce un ugual numero di simboli* a *e di simboli* b*, mantenendo per ogni forma sentenziale l'invariante* $\mid w \mid_a = \mid w \mid_b$*. A maggior ragione tale invariante, che definisce il linguaggio* L_1*, si mantiene per ogni stringa di soli terminali. Per dimostrare che* $L_1 \subseteq L(G)$*, cioè che la grammatica produce tutte le stringhe del linguaggio, procediamo per induzione sul numero* n *di* a *(e quindi di* b*) presenti nella stringa.*
Caso 0: *La stringa* λ*, di lunghezza 0, appartiene a* L_1 *e può essere prodotta da* G *applicando semplicemente la produzione* $S \to \lambda$.
Ipotesi di induzione: *Assumiamo che per* $n = k$*, tutte le stringhe di* L_1 *con numero di* a *inferiore a* k *possono essere prodotte da* G*. A tale proposito*

si osserva che esse devono necessariamente essere prodotte in derivazioni del tipo $S \Rightarrow^* \alpha S \beta \Rightarrow \alpha\beta$, cioè avendo applicato $S \rightarrow \lambda$ come ultimo passo della derivazione.

Passo di induzione: *Consideriamo le stringhe ω di L_1 con $|\omega|_a = k+1$. Esse possono essere tutte scritte nella forma $\alpha ab\beta$, o in quella $\alpha ba\beta$, per qualche stringa $\alpha\beta \in L_1$ di lunghezza $2k$. Sia $\Pi = \Theta, S \rightarrow \lambda$ la sequenza di regole applicata in una derivazione di una stringa $\alpha\beta$ di lunghezza $2k$. Vediamo ora che sequenze della forma Θ, $S \rightarrow aSbS$, $S \rightarrow \lambda$, $S \rightarrow \lambda$ producono stringhe della forma $\alpha ab\beta$. Analogo ragionamento vale per stringhe che aggiungono ba a una stringa di lunghezza $2k$. In definitiva, tutte le stringhe di ugual numero di a e b possono eesere generate da G. Quindi L_1 è libera dal contesto.*

Esercizio 2 *Si dimostri che il linguaggio $L = \{a^n b^m c^m \mid n, m \geq 0\} \cup \{a^n b^n c^m \mid n, m \geq 0\}$ è inerentemente ambiguo.*

Svolgimento 2 *Si può osservare che i due linguaggi $L_1 = \{a^n b^m c^m \mid n, m \geq 0\}$ e $L_2 = \{a^n b^n c^m \mid n, m \geq 0\}$ non sono disgiunti. La loro intersezione è infatti il linguaggio $L_3 = \{a^n b^n c^n \mid n \geq 0\}$. Questo linguaggio è infinito e non è libero dal contesto, cosìche non è possibile utilizzare tre grammatiche separate per L_3, $L \setminus L_1$ e $L \setminus L_2$. Occorre quindi che gli elementi in L_3 siano prodotti mediante le stesse regole usate per produrre qualsiasi stringa in L. Ora, data una qualsiasi grammatica per L,essa produrrà quindi un albero di derivazione t_1 per una stringa in $L \cap L_1$ e un albero di derivazione t_2 per una stringa in $L \cap L_2$. Tali alberi saranno distinti in quanto il primo deve mantenere la sincronia fra il numero di b e di c, mentre il secondo quella fra il numero di a e di b. Per le stringhe in L_3 si potranno quindi produrre entrambi gli alberi.*

Esercizio 3 *Si riduca in forma normale senza produzioni nulle e senza terminali inutili, la grammatica $G = (\{S, B, C, D\}, \{a, c, d\}, S, P)$, con l'insieme P costituito dalle regole $S \rightarrow BC$, $B \rightarrow dDc$, $B \rightarrow \lambda$, $C \rightarrow dDcB$, $C \rightarrow \lambda$, $D \rightarrow a$, $D \rightarrow \lambda$.*

Svolgimento 3 *Costruiamo in primo luogo l'insieme W dei non terminali annullabili.*

Abbiamo $W_1 = \{B, C, D\}$ come non terminali immediatamente annullabili. Per effetto della regola $S \rightarrow BC$, abbiamo poi $W_2 = \{S\}$. In definitiva si ha $W = NT$.

Possiamo allora procedere a produrre nuove regole ed eliminare le regole con lato destro λ, partendo dai non terminali in W_1. Dalla regola $B \rightarrow dDc$ otteniamo la regola $B \rightarrow dc$. Dalla regola $C \rightarrow dDcB$ otteniamo le regole $C \rightarrow dcB$, $C \rightarrow dDc$, $C \rightarrow dc$. Per i non terminali in W_2, dalla regola $S \rightarrow BC$ otteniamo $S \rightarrow C$ e $S \rightarrow B$. In definitiva si ha il nuovo insieme di regole: $P' = \{S \rightarrow BC$, $S \rightarrow B$, $S \rightarrow C$, $B \rightarrow dc$, $B \rightarrow dDc$, $C \rightarrow dcB$, $C \rightarrow dDc$, $C \rightarrow dc$, $C \rightarrow dDcB$, $D \rightarrow a\}$. Poiché si aveva originariamente $\lambda \in L$, si modifica l'insieme dei non terminali aggiungendo il nuovo non terminale Z, che diventa quindi simbolo di partenza per la grammatica e l'insieme P aggiungendo le regole $Z \to \lambda$, $Z \to BC$, $Z \to B$, $Z \to C$. Si può ora verificare che S è diventato un non terminale inutile, poiché non compare nel lato destro di nessuna produzione e si può quindi rimuoverlo insieme alle regole $S \to BC$, $S \to B$, $S \to C$. In definitiva risulta la grammatica $G' = (\{Z, B, C, D\}, \{a, c, d\}, Z, P'')$ con $P'' = \{Z \to BC$, $Z \to B$, $Z \to C$, $Z \to \lambda$, $B \to dc$, $B \to dDc$, $C \to dcB$, $C \to dDc$, $C \to dc$, $C \to dDcB$, $D \to a\}$.

Esercizio 4 Si riduca la grammatica contenente l'insieme di regole $P = \{Z \to BC$, $Z \to B$, $Z \to C$, $Z \to \lambda$, $B \to dc$, $C \to dcB$, $C \to dDc$, $C \to dc$, $D \to a\}$ in forma normale di Chomsky.

Svolgimento 4 In primo luogo introduciamo i non terminali $[B]$, $[C]$, $[D]$ e le produzioni $[B] \to b$, $[C] \to c$, $[D] \to d$. Sostituiamo quindi la regola $B \to dc$ con $B \to [D][C]$ e la regola $C \to dc$ con $C \to [D][C]$. Per portare in forma normale la regola $C \to dDc$, introduciamo ora i nuovi non terminali $[Dc]$ e $[D1]$ e sostituiamo la regola con le regole $C \to [D][Dc]$, $[Dc] \to [D1][C]$, e $[D1] \to D$. Per la regola $C \to dcB$, introduciamo invece i non terminali $[cB]$ e $[B1]$ e sostituiamo la regola con le regole $C \to [D][cB]$, $[cB] \to [C][B1]$, e $[B1] \to B$. A questo punto abbiamo l'insieme di regole $P' = \{Z \to BC$, $Z \to B$, $Z \to C$, $Z \to \lambda$, $B \to [D][C]$, $C \to [D][cB]$, $[cB] \to [C][B1]$, $[B1] \to B$, $C \to [D][Dc]$, $[Dc] \to [D1][C]$, $[D1] \to D$ $C \to [D][C]$, $[B] \to b$, $[C] \to c$, $[D] \to d$, $D \to a\}$.

Restano ora da eliminare le regole con lato destro costituito da un solo non terminale. Occorrerà sostituire tale terminale con tutti i possibili lati destri per le regole che hanno come lato sinistro il non terminale in questione. Si producono quindi le regole $Z \to [D][C]$, $Z \to [D][cB]$, $[B1] \to [D][C]$, $[D1] \to d$, $[D1] \to a$. Si ottiene quindi l'insieme $P'' = \{Z \to BC$, $Z \to [D][C]$ $Z \to [D][cB]$, $Z \to \lambda$, $B \to [D][C]$, $C \to [D][cB]$, $[cB] \to [C][B1]$, $[B1] \to [D][C]$, $C \to [D][Dc]$, $[Dc] \to [D1][C]$, $[D1] \to d$, $[D1] \to a$, $C \to [D][C]$, $[B] \to b$, $[C] \to c$, $[D] \to d$, $D \to a\}$. Infine, si osservi che il non terminale D non può essere prodotto in nessuna derivazione e può quindi essere rimosso dall'insieme dei non terminali, mentre la regola $D \to a$ può venire eliminata dall'insieme delle regole.