Equivalenza di PDA e CFG

Un linguaggio e'

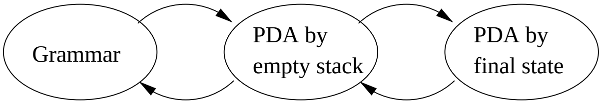
generato da una CFG

se e solo se e'

accettato da un PDA per pila vuota

se e solo se e'

accettato da un PDA per stato finale



Sappiamo gia' andare da pila vuota a stato finale (e viceversa).

Data una grammatica $G = (V, T, P, S)$, costruiamo un PDA che *simula le derivazioni*. In particolare, conviene simulare le **derivazioni sinistre** $\overset{*}{\underset{lm}{\Rightarrow}}$.

In una derivazione sinistra

$$S \underset{lm}{\Rightarrow} \gamma_1 \underset{lm}{\Rightarrow} \dots \gamma_n$$

ogni $\gamma_i = x_i A_i \alpha_i$ e' una *forma sentenziale sinistra*:

1. $x_i \in T^*$ e' una *stringa di terminali*, $A_i \in V$ e' la prima variabile a sinistra e $\alpha_i \in (V \cup T)^*$ e' la *coda*, ovvero la stringa di terminali e non terminali ancora da espandere
2. inoltre, viene applicata una regola della grammatica ad A_i, ovvero $A_i \to \beta_i \in P$ e $\gamma_{i+1} = x_i \beta_i \alpha_i$

Idea della simulazione

Per simulare una derivazione $S \underset{lm}{\overset{*}{\Rightarrow}} xA\alpha$ l'automa si trovera' nella seguente configurazione, avendo letto la stringa di input x

$$(q, y, A\alpha)$$

dove y e' l'input ancora da leggere e la pila memorizza la variabile A (al top) e la coda della produzione α.
Intuitivamente l'automa deve eseguire mosse che permettono di derivare y da $A\alpha$.

L'automa puo' eseguire le seguenti mosse:

- espandere la variabile A al top della pila, applicando una produzione $A \rightarrow \beta \in P$. Di fatto esegue una ϵ-transizione in

$$(q, y, \beta\alpha)$$

- se β e' a sua volta della forma $B\beta'$, allora continua ad applicare regole. Altrimenti, se $\beta = w\beta'$ inizia in una stringa di terminali, allora l'automa li deve eliminare, se li legge in input.

Se tutte le scommesse sono giuste, il PDA finisce l'input con la pila vuota.

Definizione Formale

Sia $G = (V, T, Q, S)$ una CFG. Definiamo il PDA come

$$P_G = (\{q\}, T, V \cup T, \delta, q, S),$$

dove

- per ogni variabile $A \in V$,

$$\delta(q, \epsilon, A) = \{(q, \beta) : A \to \beta \ \in \ Q\},$$

- per ogni terminale $a \in T$

$$\delta(q, a, a) = \{(q, \epsilon)\},$$

Teorema

Sia $G = (V, T, Q, S)$ una CFG, e sia P_G il PDA definito nel lucido precedente. Allora, per ogni $w \in T^*$, abbiamo

$$w \in L(G) \Leftrightarrow w \in N(P_G)$$

Prova Per provare che i due linguaggi coincidono dobbiamo fare vedere che, per ogni w,

$$S \underset{lm}{\overset{*}{\Rightarrow}} w \text{ sse } (q, w, S) \vdash^* (q, \epsilon, \epsilon)$$

Da Derivazione ad automa

Se $S \underset{lm}{\overset{*}{\Rightarrow}} w$ allora per definizione di derivazione sinistra esiste

$$S = \gamma_1 \underset{lm}{\Rightarrow} \gamma_1 \underset{lm}{\Rightarrow} \gamma_2 \ldots \underset{lm}{\Rightarrow} \gamma_n = w$$

in cui ogni γ_i e' forma sentenziale sinistra.
Proviamo per induzione sul numero di passi di derivazione che

$$(q, w, S) \vdash^* (q, y_i, \alpha_i)$$

dove $\gamma_i = x_i \alpha_i$ (α_i e' la coda) e y_i e' la stringa tale che $w = x_i y_i$ (x_i e' la parte gia' letta e y_i quella ancora da leggere)

- **Caso Base**: Per $i = 1$ abbiamo $\gamma_1 = S$ e $x_i = \epsilon$ e $y_i = w$. Banalmente $(q, w, S) \vdash^* (q, w, S)$
- **Caso Induttivo**: Per $i < n$ abbiamo

$$(q, w, S) \vdash^* (q, y_i, \alpha_i)$$

dove α_i e' la coda dell'i-esimo passo di derivazione, ovvero $\alpha_i = A\beta_i$ per una qualche variabile A.
Nell'ultimo passo della derivazione $\gamma_i \underset{lm}{\Rightarrow} \gamma_{i+1}$ deve essere applicata una regola della grammatica ad A, suppoiniamo $A \to \beta_i' \in P$.
L'automa e' nella configurazione (q, y_i, α_i), quindi A e' al top della pila. Possiamo applicare la regola e trovare

$$(q, y_i, \alpha_i) \vdash (q, y_i, \beta_i'\beta)$$

Se β_i' contiene terminali possiamo leggerli e consumare l'input y_i fino a che il contenuto della pila non inizia con un non terminale. Avremo

$$(q, y_i, \beta_i'\beta) \vdash (q, y_{i+1}, \gamma_{i+1})$$

che rappresenta la successiva forma sentenziale sinistra.
Notiamo che nel caso al passo n abbiamo $\alpha_n = \epsilon$ (la coda e' vuota), quindi la pila e' vuota

$$(q, w, S) \vdash (q, \epsilon, \epsilon)$$

e la stringa w viene accettata per pila vuota.

Da Automa a Derivazione

Bisogna fare vedere un fatto piu' generale: se $(q, x, A) \vdash^* (q, \epsilon, \epsilon)$ allora $A \underset{lm}{\Rightarrow}^* x$ per ogni variabile A.

In particolare la proprieta' vale per il simbolo iniziale della grammatica S.

Per induzione sul numero di mosse fatte dall'automa.

- **Caso Base**: Dato che A e' al top della pila l'automa fa una mossa che corrisponde all'applicazione di una produzione. In particolare deve essere $A \rightarrow \epsilon \in P$. Di conseguenza $x = \epsilon$ e $A \underset{lm}{\Rightarrow}^* \epsilon$

Passo Induttivo

L'automa deriva $(q, x, A) \vdash^* (q, \epsilon, \epsilon)$ in n mosse con $n > 1$. Come prima la prima mossa deve essere l'applicazione di una produzione del tipo $A \rightarrow \beta \in P$. Quindi

$$(q, x, A) \vdash (q, x, \beta) \vdash^* (q, \epsilon, \epsilon)$$

Supponiamo che $\beta = X_1 \dots X_k$ dove X_j e' una variabile o un terminale.
Le $n - 1$ mosse successive devono eliminare $X_1 \dots X_k$ dalla pila (arriviamo alla pila vuota). Dato che alla fine viene consumato x, allora deve essere possibile scomporre x come

$$x = x_1 \dots x_k$$

Passo Induttivo

Dato $x = x_1 \dots x_k$ e la forma della pila, per ogni $i \in \{1, k\}$, x_i e' la parte dell'input consumata fino a quando si elimina il simbolo X_i dalla pila. Deve essere

$$(q, x_i \dots x_k, X_i) \vdash^* (q, x_{i+1} \dots x_k, \epsilon)$$

Quindi, per ogni $i \in \{1, k\}$ abbiamo una derivazione $X_i \underset{lm}{\Rightarrow}^* x_i$ (se X_i e' un nonterminale per I.I.)
Costruiamo quindi la derivazione sinistra

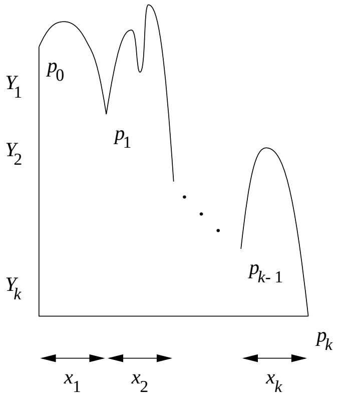
$$AX_i \underset{lm}{\Rightarrow} X_1 \dots X_k \underset{lm}{\Rightarrow}^* x_1 X_{i+1} \dots X_k \underset{lm}{\Rightarrow}^* \dots x_1 \dots x_k = x$$

Da PDA a CFG

La costruzione che permette di ottenere una grammatica da un automa e' piu' complicata.
Si basa sulla seguente idea intuitiva: l'evento cruciale nell'elaborazione di un PDA consiste nell'eliminazione di un simbolo dalla pila, conseguente alla lettura di una parte di input.
eliminazione: e' l'effetto finale di una o piu' mosse (alla fine la pila e' vuota). Se viene eliminata la variabile Y in questo processo verra' letta una parte di input y. Alla fine la pila e' vuota.

Da PDA a CFG

Vediamo come un PDA consuma $x = x_1 x_2 \cdots x_k$ e vuota la pila.



Dallo stato p_i, eliminando Y_i dalla pila, l'automa si muove in p_{i+1}. L'effetto di queste mosse e' quello di leggere x_i

Definizione Intuitiva

- Definiremo una grammatica con variabili della forma $[p_{i-1} Y_i p_i]$ che rappresentano il passaggio dallo stato p_{i-1} allo stato p_i con l'effetto di eliminare Y_i.
- Una variabile di questo tipo dovrebbe generare tutte le stringhe w che portano l'automa dallo stato p_{i-1} allo stato p eliminando Y_i dalla pila.

Definizione Formale

Sia $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$ un PDA. Definiamo $G = (V, \Sigma, R, S)$, dove

$$V = \{[pXq] : \{p, q\} \subseteq Q, X \in \Gamma\} \cup \{S\}$$

Le variabili sono del tipo $[pXq]$ oltre ad un nuovo simbolo iniziale S
Le produzioni R sono di due tipi.

Primo Tipo di Regole

Per ogni stato $p \in Q$, abbiamo la produzione

$$\{S \rightarrow [q_0 Z_0 p] : p \in Q\}$$

Il simbolo $[q_0 Z_0 p]$ rappresenta tutte le stringhe w che provocano l'eliminazione di Z_0 dalla pila, quando l'automa si muove dallo stato iniziale q_0 a p. Sono le stringhe accettate per pila vuota, partendo dalla configurazione iniziale.

Per ogni stato $q \in Q$, supponiamo che $\delta(q, a, X)$ contenga $(r, X_1, \ldots, X_k)$. Allora per ogni sequenza di stati $\{r_1, \ldots, r_k\} \subseteq Q$ mettiamo una produzione

$$[\mathbf{q} X r_k] \to a[r Y_1 r_1] \ldots [r_{k-1} Y_k r_k]$$

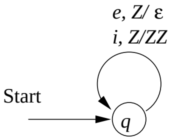
La regola

$$[\mathbf{q}Xr_k] \to a[rY_1r_1]\dots[r_{k-1}Y_kr_k]$$

dice che per muoversi da q ad r_k eliminando X un modo puo' essere: leggere a e muoversi in r, poi eliminare Y_1 dalla pila, e muoversi in r_1. E cosi' via fino ad r_k.
Nota: $a \in \Sigma \cup \{\epsilon\}$ e k potrebbe anche esere 0. In tal caso la mossa dell'automa e' (r, ϵ).

Esempio

Convertiamo



$$P_N = (\{q\}, \{i, e\}, \{Z\}, \delta_N, q, Z),$$

dove $\delta_N(q, i, Z) = \{(q, ZZ)\}$,
e $\delta_N(q, e, Z) = \{(q, \epsilon)\}$ in una grammatica

$$G = (V, \{i, e\}, R, S),$$

dove $V = \{[qZq], S\}$, and
$R = \{[qZq] \rightarrow i[qZq][qZq], [qZq] \rightarrow e\}$.
Se rimpiazziamo $[qZq]$ con A otteniamo le produzioni $S \rightarrow A$ e $A \rightarrow iAA|e$.

Esempio

Convertiamo $P = (\{p, q\}, \{0, 1\}, \{X, Z_0\}, \delta, q, Z_0)$, dove δ e' data da

1. $\delta(q, 1, Z_0) = \{(q, XZ_0)\}$
2. $\delta(q, 1, X) = \{(q, XX)\}$
3. $\delta(q, 0, X) = \{(p, X)\}$
4. $\delta(q, \epsilon, X) = \{(q, \epsilon)\}$
5. $\delta(p, 1, X) = \{(p, \epsilon)\}$
6. $\delta(p, 0, Z_0) = \{(q, Z_0)\}$

in una CFG.

Otteniamo $G = (V, \{0,1\}, R, S)$, dove

$$V = \{[pXp], [pXq], [pZ_0p], [pZ_0q], S\}$$

e le produzioni in R sono

$S \to [qZ_0q] | [qZ_0p]$

Dalla regola (1):

$[qZ_0q] \to 1[qXq][qZ_0q]$

$[qZ_0q] \to 1[qXp][pZ_0q]$

$[qZ_0p] \to 1[qXq][qZ_0p]$

$[qZ_0p] \to 1[qXp][pZ_0p]$

Dalla regola (2):

$[qXq] \to 1[qXq][qXq]$

$[qXq] \to 1[qXp][pXq]$

$[qXp] \to 1[qXq][qXp]$

$[qXp] \to 1[qXp][pXp]$

Dalla regola (3):
$[qXq] \rightarrow 0[pXq]$
$[qXp] \rightarrow 0[pXp]$
Dalla regola (4):
$[qXq] \rightarrow \epsilon$
Dalla regola (5):
$[pXp] \rightarrow 1$
Dalla regola (6):
$[pZ_0q] \rightarrow 0[qZ_0q]$
$[pZ_0p] \rightarrow 0[qZ_0p]$

Sia $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$ un PDA, e $G = (V, \Sigma, R, S)$, la grammatica definita come prima. Allora, per ogni $w \in T^*$, abbiamo

$$w \in L(G) \Leftrightarrow w \in N(P)$$

Prova Per provare che i due linguaggi coincidono dobbiamo fare vedere che, per ogni w, e per ogni coppia di stati $p, q \in Q$ e per ogni $X \in \Gamma$,

$$[qXp] \underset{lm}{\overset{*}{\Rightarrow}} w \text{ sse } (q, w, X) \vdash^* (p, \epsilon, \epsilon)$$

L'interpretazione del significato delle variabili $[qXp]$ e' corretto

Supponiamo che $(q, w, X) \vdash^* (p, \epsilon, \epsilon)$. Per induzione sul numero di mosse dell'automa facciamo vedere che $[qXp] \underset{lm}{\overset{*}{\Rightarrow}} w$.

Caso Base: La pila viene svuotata in un passo. Allora deve essere $\delta(q, w, X) = (p, \epsilon)$ e $w \in \Sigma \cup \{\epsilon\}$. Per definizione delle produzioni abbiamo $[qXp] \to w \in R$.

Caso Induttivo

Supponiamo che $(q, w, X) \vdash^* (p, \epsilon, \epsilon)$ richieda n passi con $n > 1$.
Esaminando la prima mossa avremo

$$(q, w, X) \vdash (r_0, x, Y_1 \dots Y_k) \vdash^* (p, \epsilon, \epsilon)$$

per qualche $r_0 \in Q$, $a \in \Sigma \cup \{\epsilon\}$ e $Y_1, \dots, Y_k \in \Gamma$. Deve essere $w = ax$ e $(r_0, Y_1 \dots Y_k) \in \delta(p, a, X)$.
Allora, per definizione delle produzioni avremo:

$$[qXr_k] \rightarrow a[r_0 Y_1 r_1] \dots [r_{k-1} Y_k r_k]$$

dove $r_k = p$ e $r_1 \dots r_k$ sono stati arbitari.

Caso Induttivo

Esaminando la sequenza di mosse

$$(r_0, x, Y_1 \dots Y_k) \vdash^* (p, \epsilon, \epsilon)$$

vediamo che vengono eliminati tutti i simboli dalla pila $Y_1, \dots, Y_k \in \Gamma$.
Allora, deve essere $x = x_1, \dots x_k$ dove x_i e' la porzione dell'input letta, quando viene eliminato Y_i, e l'automa si muove da r_{i-1} a r_i.
Formalmente, per ogni $i \in \{1, k\}$,

$$(r_{i-1}, x_i, Y_i) \vdash^* (r_i, \epsilon, \epsilon)$$

Per ipotesi induttiva abbiamo

$$[r_{i-1}] Y_i r_i \underset{lm}{\overset{*}{\Rightarrow}} x_i$$

Caso Induttivo

Unendo le derivazioni

$$[r_{i-1}]Y_i r_i \underset{lm}{\overset{*}{\Rightarrow}} x_i$$

con la produzione

$$[qXr_k] \rightarrow a[r_0 Y_1 r_1] \ldots [r_{k-1} Y_k r_k]$$

otteniamo

$$[qXr_k] \underset{lm}{\overset{*}{\Rightarrow}} ax_1 \ldots x_k = w$$

Da derivazione ad automa

Supponiamo che $[qXp] \overset{*}{\underset{lm}{\Rightarrow}} w$. Per induzione sul numero di passi di derivazione facciamo vedere che $(q, w, X) \vdash^* (p, \epsilon, \epsilon)$.
Caso Base: $[qXp] \rightarrow w \in R$. Per definizione delle regole deve essere eliminato X da q, muovendosi in p, e leggendo a. L'automa ha la mossa, $(q, \epsilon) \in \delta(q, a, X)$. Quindi $(q, w, X) \vdash (p, \epsilon, \epsilon)$.

Caso Induttivo

Supponiamo che $[qXp] \underset{lm}{\overset{*}{\Rightarrow}} w$ richieda n passi con $n > 1$.
Esaminando la prima mossa avremo

$$[qXr_k] \rightarrow a[r_0 Y_1 r_1] \dots [r_{k-1} Y_k r_k] \underset{lm}{\overset{*}{\Rightarrow}} w$$

dove $r_k = p$ e gli stati $r_1 \dots r_k$ sono stati tali che $(r_0, Y_1, \dots, Y_k) \in \delta(q, a, X)$.
Esaminado la derivazione e' chiaro che possiamo scomporre $w = aw_1 \dots w_k$ dove per ogni i

$$[r_{i_1} Y_i r_i] \underset{lm}{\overset{*}{\Rightarrow}} w_i$$

Inoltre per ipotesi induttiva abbiamo, per ogni i,

$$(r_{i-1}, w_i, Y_i) \vdash (r_i, \epsilon, \epsilon)$$

Mettendo insieme tutte le mosse dell'automa otteniamo

$$(q, aw_1 \dots w_k, X) \vdash (r_0, w_1 \dots w_k, Y_1 \dots Y_k) \vdash^* (r_k, \epsilon, \epsilon).$$

Abbiamo provato che

$$[qXp] \underset{lm}{\overset{*}{\Rightarrow}} w \text{ sse } (q, w, X) \vdash^* (p, \epsilon, \epsilon)$$

Da questo dobbiamo dedurre che $L(G) = N(P)$.
Notiamo che $w \in L(G)$ sse $S \underset{lm}{\overset{*}{\Rightarrow}} w$, dal simbolo iniziale S.
Tuttavia, per S abbiamo solo regole del tipo

$$S \rightarrow [q_0 Z_0 p]$$

Quindi, se $S \underset{lm}{\overset{*}{\Rightarrow}} w$ sse $[q_0 Z_0 p] \underset{lm}{\overset{*}{\Rightarrow}} w$ sse $(q_0, w, Z_0) \vdash^* (p, \epsilon, \epsilon)$.
Sono esattamente le stringhe accettate dall'automa partendo dalla configurazione iniziale, per pila vuota.

PDA deterministici

- definiremo DPDA (il sottocaso di automa a pila deterministico)
- la classe dei linuguaggi riconosciuti da DPDA si colloca tra i linguaggi regolari ed i CFL
- questa classe di linguaggi e' importante per comprendere i costrutti dei linguaggi di programmazione

PDA deterministici

Un PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ e' *deterministico* se e solo se

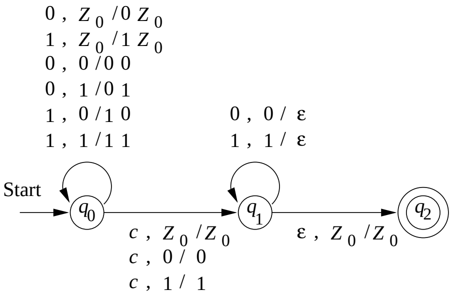
1. $\delta(q, a, X)$ ha al massimo un elemento
2. Se $\delta(q, a, X)$ non e' vuoto, allora $\delta(q, \epsilon, X)$ deve essere vuoto.

Per ogni stato deve esserci al piu' una mossa, per ogni simbolo a e simbolo sulla pila X. Inoltre non deve esserci conflitto tra la lettura di un input ed una ϵ-transizione

Definiamo

$$L_{wcwr} = \{wcw^R : w \in \{0,1\}^*\}$$

Allora L_{wcwr} e' riconosciuto dal seguente DPDA



Mostreremo che Regolari$\subset L(\mathrm{DPDA}) \subset$ CFL

Teorema 6.17: Se L e' regolare, allora $L = L(P)$ per qualche DPDA P.

Prova: Dato che L e' regolare, esiste un DFA A tale che $L = L(A)$. Sia

$$A = (Q, \Sigma, \delta_A, q_0, F)$$

definiamo il DPDA

$$P = (Q, \Sigma, \{Z_0\}, \delta_P, q_0, Z_0, F),$$

dove

$$\delta_P(q, a, Z_0) = \{(\delta_A(q, a), Z_0)\},$$

per tutti i $p, q \in Q$ e $a \in \Sigma$.

Un'induzione su $|w|$ ci da'

$$(q_0, w, Z_0) \overset{*}{\vdash} (p, \epsilon, Z_0) \Leftrightarrow \hat{\delta}_A(q_0, w) = p$$

E i DPDA che accettano per pila vuota?

- Possono riconoscere solo CFL con la proprieta' del prefisso.
- Un linguaggio L ha la *proprieta' del prefisso* se non esistono due stringhe distinte in L, tali che una e' un prefisso dell'altra.
- Esempio: L_{wcwr} ha la proprieta' del prefisso.
  Siano $w_1 = xcx^R$ e $w_2 = ycy^R$ stringhe del linguaggio e w_2 prefisso di w_1. Allora w_2 e' piu' breve di w_1, quindi la c di w_2 e' in una posizione dove w_1 ha uno 0 o un 1.
- Esempio: $\{0\}^*$ non ha la proprieta' del prefisso.

- Abbiamo visto che Regolari $\subseteq L(\mathrm{DPDA})$.
- $L_{wcwr} \in L(\mathrm{DPDA}) \setminus$ Regolari
- Ci sono linguaggi in CFL$\setminus L(\mathrm{DPDA})$.
  Si, per esempio

$$L_{wwr} = \{ww^R \mid w \in \{0,1\}^*\}.$$

Il linguaggio e' simile a quello visto in precedenza. Non esiste un DPDA, in questo caso non e' possibile stabilire quando finisce w ed inizia w^R.

Cosa possiamo dire su DPDA e grammatiche ambigue?

L_{wwr} ha una grammatica non ambigua

$$S \rightarrow 0S0|1S1|\epsilon$$

ma non e' $L(\mathrm{DPDA})$.

Per l'inverso abbiamo
Teorema 6.20: Se $L = N(P)$ per qualche DPDA P, allora L ha una CFG non ambigua.
Prova: Guardando la prova del teorema 6.14 vediamo che se la costruziuone e' applicata ad un DPDA, il risultato e' un CFG con derivazioni a sinistra uniche per ogni stringa.

Il teorema 6.20 puo' essere rafforzato:

Teorema 6.21: Se $L = L(P)$ per qualche DPDA P, allora L ha una CFG non ambigua.

Prova:

- Sia \$ un simbolo fuori dell'alfabeto di L, e sia $L' = L\$$.
- E' facile vedere che L' ha la proprieta' del prefisso.
- Per il teorema 6.20 abbiamo che $L' = N(P')$ per qualche DPDA P'.
- Per il teorema 6.20 $N(P')$ puo' essere generato da una CFG G' non ambigua
- Modifichiamo G' in G, tale che $L(G) = L$, aggiungendo la produzione

$$\$ \rightarrow \epsilon$$

- Dato che G' ha derivazioni a sinistra uniche, anche G' le ha, dato che l'unica cosa nuova e' l'aggiunta di derivazioni

$$w\$ \underset{lm}{\Rightarrow} w$$

alla fine.