

TC CURS 13

AUTOMATE PUSH DOWN DETERMINISTE

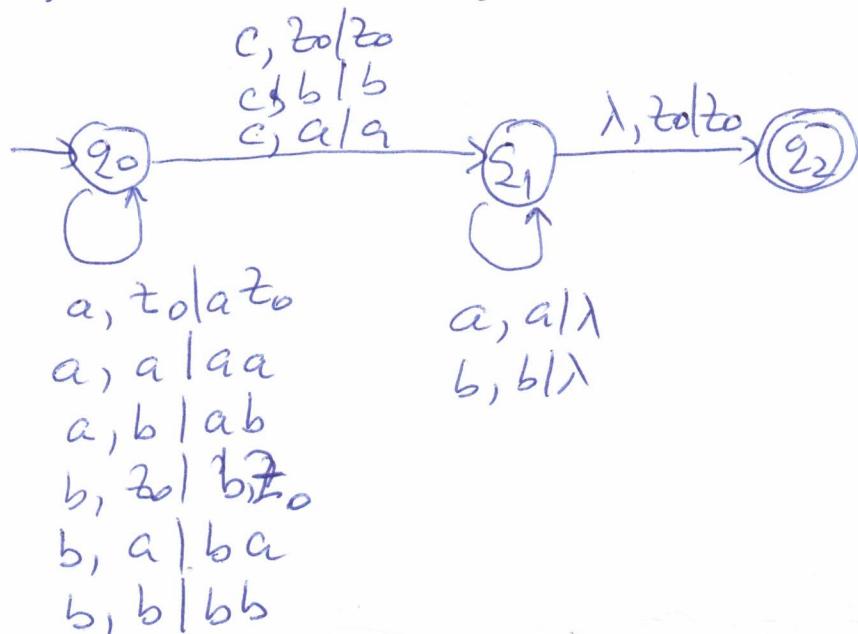
Def Spunem că APD $A = (Q, \Sigma, P, \delta, \Sigma_0, z_0, F)$ este determinist dacă $\forall q \in Q, \forall a \in \Sigma, \forall z \in \Gamma$

$$|\delta(q, a, z)| + |\delta(q, \lambda, z)| \leq 1$$

Cu alte cuvinte, atunci când automatul A este în starea z , iar în vîrful stivei are simbolul z , există cel mult o tranziție a lui A din Q : fie o transiție etichetată cu $a \in \Sigma$, fie o λ -transiție. Aceasta înseamnă că schimbările de configurație pe care A le face sunt unice.

Exemplu:

1) Automatul A_1 :

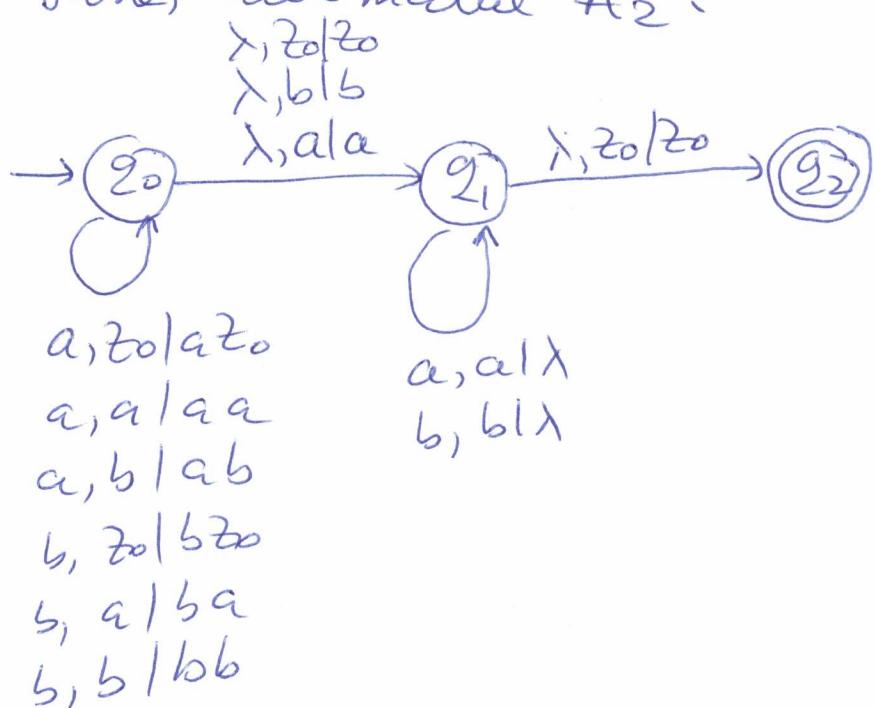


Este un automat determinist care recunoaște limbajul:

$$L(A_1) = \{ w \in \{a, b\}^* \mid w \text{ este posternatul lui } w \}$$

(de exemplu, pentru $w = aabb, w' = bbbaa$)

2) Jumătate, automatul A_2 :



Care recunoaște $L(A_2) = \{ww' \mid w \in \{a, b\}^*\}, w'$ este posternatul lui w

Este determinist, deoarece, de exemplu, din starea q_0 , arând a în vârful stivei și b următorul simbol de pe lărgul de intrare, fie poate să rămână în q_0 și să avansese la următorul simbol din intrare, punând b pe stivă, fie poate să treacă la starea q_1 lăsând stiva nechimbată și fără să avansese la următorul simbol din intrare (λ -transiție).

Nedeterminismul lui A_2 apare doar în sefăru că nu știm care este "granița" dintre

$w \in w'$ (de fapt $\overset{=3=}{w}$ și w' , pentru $M_1 = M_1^1$), pe cind în casul lui A_1 , mijlocul sănătății acceptat este rezultat de mișcările c .

3). Să considerăm $L = \{a^n b^n / n \geq 1\} \cup \{a^n b^{2n} / n \geq 1\}$.

Anotăvă că L nu poate fi acceptat de niciun APD determinist.

Preneperem pră alătură că L este acceptat de APD determinist $A_3 = (Q, \{a, b\}, \Gamma, \delta, q_0, z_0, F)$.

A_3 nu poate să accepte pe L ceci violarea stivei, deoarece pentru simbol $a^n b^{2n}$, $n \geq 1$ dat, după ce A_3 citează prefixul $a^n b^n$, care este în L , automatul videază stiva și deci nu mai poate continua să citească restul simulei (b^n), aceasta înținând cont de faptul că transițiile în A_3 sunt unice. Rezultă că $F \neq \emptyset$, adică A_3 acceptă pe L cu stări finale.

Pentru o multime M să notăm cu \bar{M} o copie a sa, adică $\bar{M} = \{\bar{a} \mid a \in M\}$.

Pornind de la A_3 vom construi APD A'_1 în felul următor.

Așa cănd A_3 trece din starea z_0 într-o stare finală q , citind un simbol de formă $a^i b^i$, $i \geq 1$, rezultă că din q mai există transiții către, deoarece mișcările lui A_3 sunt unic determinante.

= 4 =

? pentru a asigura că și contextul a^ib^{2i} este acceptat de A_3 . În A' vom face ca toate transițiile cu ' b ' ce punesc din stări finale năle să se transforme în transiții cu ' c ', ce vor aduce într-o copie a stării în care ar fi trecut A_3 dacă ar fi acceptat ' b '. În momentul în care A' ajunge într-o stare de forma $\bar{z}, q \in Q$, toate transițiile cu ' b ' din \bar{z} vor fi transformate în transiții cu ' c '. Este ca și cum am lucra într-o copie a lui A_3 , în care avem doar transiții cu ' c '. Astfel:

$$A' = (Q \cup \bar{Q}, \{a, b, c\}, \Gamma, \delta', q_0, z_0, F \cup \bar{F})$$

$$\delta'(\bar{z}, a, z) = \delta(z, a, z), \forall q \in Q, \forall z \in \Gamma$$

$$\delta'(\bar{z}, b, z) = \delta(z, b, z), \forall q \in Q - F, \forall z \in \Gamma$$

$$\delta'(\bar{z}, c, z) = (\bar{p}, \alpha), \text{ și astfel încât } \delta(z, b, z) = (p, \alpha)$$

$$\delta'(\bar{z}, c, z) = (\bar{r}, \beta), \text{ și } q, r \in Q \text{ astfel încât } \delta(z, b, z) = (r, \beta)$$

În felul acesta, limboajul acceptat de A' va fi $L(A') = \{a^n b^n c^n \mid n \geq 1\}$, care nu este un limboaj independent de context, contradicție.

Definiție: Spunem că un limboaj independent de context este determinist dacă este acceptat de un automat push down determinist.

=5=
4) Limbajul $L_{pal} = \{ w w' \mid w \in \{a, b\}^*, w' \text{ este netul lui } w \}$ al sirurilor palindromice de lungime pară nu poate fi acceptat de niciun automat pushdown determinist. Se poate demonstra acest lucru printr-un ratiuneal axemontor ca în exemplul 3.

L_{pal} poate fi iesit generat de gramatica neambigă cu producții:

$$S \rightarrow a S a \quad b S b \quad \lambda$$

OBSERVAȚII

- 1) Automatele pushdown deterministe au un rol esențial în ceea ce privește majoritatea compilatoarelor diverselor limbaje de programare sunt implementate folosind automate pushdown deterministe.
- 2) Pentru orice automat pushdown determinist se poate construi o gramatică independentă de context neambigă echivalentă.
- 3) Reciproce afirmații 2 nu este cedevică, are cum am văzut în exemplul 4: există limbaje independente de context neambigue care nu sunt determinante.

- 4) Problema echivalenței a două APD deterministe
oarecare este decidabilă (există un algoritm
care decide dacă două APD deterministe arbitraze
recunosc același limbaj (Gereud Séniergues, 1997))
5) Problema echivalenței a două APD oarecare
(neDeterministe) nă este decidabilă.

PROBLEMA CORESPONDENȚEI LUI POST (PCP)

Se dau două liste de simboluri peste Σ , $| \Sigma | \geq 2$.

$$A = \{x_1, x_2, \dots, x_k\}, \quad x_1, \dots, x_k \in \Sigma^*$$

$$B = \{y_1, y_2, \dots, y_k\}, \quad y_1, \dots, y_k \in \Sigma^*$$

Să spunem că avem o soluție pentru PCP dacă
există o secvență de numere i_1, i_2, \dots, i_n , $n \geq 1$,
 $1 \leq i_1, i_2, \dots, i_n \leq k$, astfel încât

$$x_{i_1} x_{i_2} \dots x_{i_n} = y_{i_1} y_{i_2} \dots y_{i_n}$$

$$\text{Exemplu: 1) } A = \{1, 10111, 10\}, \quad B = \{111, 10, 0\}.$$

O soluție pentru A și B este: 2, 1, 1, 3

$$A: \underline{10111} \quad \underline{1} \quad \underline{10}$$

$$B: \underline{10} \quad \underline{111} \quad \underline{111} \quad \underline{0}$$

$$2) \quad A = \{10, 011, 101\} \quad B = \{101, 11, 011\}$$

Nu are soluție

Teorema PCP nu este decișibilă, cu alte cuvinte nu există un algoritm care să decida dacă PCP are sau nu reie.

Demonstratia acestei teoreme se poate face cu ajutorul mașinilor Turing. Pentru $|\Sigma|=1$, PCP este decișibilă.
Vom utiliza PCP pentru a demonstra că există anumite proprietăți care nu sunt decișibile în cazul gramaticilor independente de context.

PROBLEME DE DECIZIE PENTRU GRAMATICILE

INDEPENDENTE DE CONTEXT

Teorema 1 (Problema apartenenței)

Se poate decide algoritmic dacă pentru o g.i.c. $G=(N, \Sigma, S, P)$ arbitrară și pentru $w \in \Sigma^*$, w este în $L(G)$ sau nu.

Au demonstrat acesta într-un curs anterior.

După ce am simplificat gramatica, orice producție a gramaticii va introduce cel puțin un simbol (terminal sau neterminál) nou în derivarea curentă (excepție fiind $S \rightarrow \lambda$ atunci când $\lambda \in L(G)$, ceea ce înseamnă că S nu mai apere în membrul drept al micului producții).

= 8 =

- Există foarte mulți algoritmi care rezolvă problema apărută în teorie. Printre acești se numără:
- algoritmul CYK (Cocke-Yates-Kasami) care pornește de la gramatică în formă normală Chomsky
 - algoritmul Earley, pentru limbișe naturale
 - algoritmi de tip LL, băsiți pe gramatici de tip LL, care descriu sintaxe limbișelor Pascal sau like-Pascal
 - algoritmi de tip LR, băsiți pe gramatici de tip LR(1), care descriu sintaxe unei limbișe de programare ca C, C++, Java etc.

Teorema 2 Nu se poate decide dacă o gramatică independentă de context este ambiguă.

Demonstratie Fie $A = \{x_1, x_2, \dots, x_K\}$, $B = \{y_1, \dots, y_K\}$, $x_1 \rightarrow x_K, y_1 \rightarrow y_K \in \Sigma^*$ și iertantă a PCP.

Considerăm limbajele:

$$L_A = \{x_{i_1} x_{i_2} \dots x_{i_n} \$ 0^i_1 0^i_2 \dots 0^i_n \mid 1 \leq i_1, \dots, i_n \leq K, n \geq 0\}$$

$$L_B = \{y_{i_1} y_{i_2} \dots y_{i_n} \$ 0^{i_1} 0^{i_2} \dots 0^{i_n} \mid 1 \leq i_1, \dots, i_n \leq K, n \geq 0\},$$

unde $0, 1, \$ \notin \Sigma$.

= g =

În gramatica $G = (\{S, S_A, S_B\}, \Sigma_{0301}, \$, S, P)$

unde $P = \{ S \rightarrow S_A | S_B \}$

$S_A \rightarrow x_i S_A 01^i | \$, i=1 \dots k$

$S_B \rightarrow y_i S_B 01^i | \$, i=1 \dots k$

}

Evident, G este independentă de context și

$L(G) = L_{AULB}$.

Înstanta PCP(A,B) are soluție $\Leftrightarrow \exists i_1 \dots i_m$

astfel încât $x_{i_1} \dots x_{i_m} = y_{i_1} \dots y_{i_m}, n \geq 1, 1 \leq i_1 \dots i_m \leq k$.

În G avem derivările lungi obținute:

$S \Rightarrow S_A \Rightarrow x_{i_1} S_A 01^{i_1} \Rightarrow \dots \Rightarrow x_{i_1} \dots x_{i_m} \$ 01^{i_1} \dots 01^{i_m}$

$S \Rightarrow S_B \Rightarrow y_{i_1} S_B 01^{i_1} \Rightarrow \dots \Rightarrow y_{i_1} \dots y_{i_m} \$ 01^{i_1} \dots 01^{i_m}$

pentru acelorași $i_1 \dots i_m$, $x_{i_1} \dots x_{i_m} \$ 01^{i_1} \dots 01^{i_m} =$

$= y_{i_1} \dots y_{i_m} \$ 01^{i_1} \dots 01^{i_m}$, deci dacă PCP(A,B) are

soluție, atunci G este ambiguă.

Reciproc, având că dacă G este ambiguă, atunci PCP(A,B) are soluție

Dovorește să răstreacă forma 01^i de la dreapta
lui $\$$ ne spune ce producție s-are aplicat,
rezultă că din S_A sau din S_B avem

plerivari (stăngi) unice pentru un zis det.

Rezultă că ambiguitatea este produsă de aplicarea productiilor $S \rightarrow S_A / S_B$.

Ce alte ceeaună, există un $z \in L_A \cup L_B$
astfel încât $S_A \xrightarrow{*} z_{AB}$, $S_B \xrightarrow{*} z_{AB}$, deci
există i, j , astfel ca $x_1 \dots x_n = y_i - j_n$.

Rezultă că $L(G)$ ambiguă \Leftrightarrow PCP(A, B) are
soluție, care este o problemă undecidibilă.

Teorema 3 Fie G_1 și G_2 două gramatici independente
de context, R expresie regulată.

Următoarele probleme sunt undecidibile:

a) G_1 este ambiguă (veri Teorema 2)

b) $L(G_1) \cap L(G_2) = \emptyset$

c) $L(G_1) = L(G_2)$

d) $L(G_1) = L(R)$

e) $L(G_1) = \Sigma^*$, unde Σ este alfabet

f) $L(G_2) - L(G_1) = \emptyset$

g) $L(R) - L(G_1) = \emptyset$

Teorema 4 Fie A_1 și A_2 două APD. Este undecidibil
deci $L(A_1) = L(A_2)$.

(Pentru APD determinante, problema este decidibilă)

= 11 =

IERARHIA LUI CHOMSKY

Să ne reamintim definiția (ceo mai generală) a unei grămatice formale (Chomsky).

$$G = (N, \Sigma, S, P)$$

- N alfabetul neterminalelor
- Σ alfabetul terminalilor
- S ∈ N simbolul de start

- P multimea producțiilor de forma

$$\alpha \rightarrow \beta, \alpha \in (N \cup \Sigma)^*, N(N \cup \Sigma)^*, \beta \in (N \cup \Sigma)^*$$

GRAMATICI SI LIMBAJE DE TIPOUL 0

Grămaticile ale căror producții sunt în forme ceo mai generale (ce mai sus, fără nicio restricție) se numesc grămatici de tipul 0, iar limbajele generate de aceste grămatice formează familia limbajelor de tipul 0, notată cu L_0 .

Limbajele de tipul 0 sunt acceptate de Mașinile Turing. Se mai numesc și limbaje Turing-acceptabile:

Exemplu $S \rightarrow aSAB | bb$

$$aSA \rightarrow a$$

$$B \rightarrow bab$$

= 12 =

De asemenea, limbajele de tipul 0 se mai numesc și limbaje recursive enumerabile.

Observatie. Există limbaje care nu sunt recursive enumerabile!

GRAMATICI SI LIMBAJE DE TIPUL 1

Definiție 1 O gramatică Chomsky $G = (N, \Sigma, S, P)$ numește gramatică dependentă de context dacă produsurile sale sunt de forma:

$$x A y \rightarrow x z y, A \in N, x, y \in (N \cup \Sigma)^*, z \in (N \cup \Sigma)^+$$

(neterminalul A este înlocuit de z între prezenta contextelor x și y)

Dacă vrem ca $\lambda \in L(G)$, atunci este惟ine să producția $S \rightarrow \lambda$ ce condiția ca S să nu apară în membrul drept al niciunei producții.

Definiție 2 O gramatică Chomsky $G = (N, \Sigma, S, P)$ se numește monotonă dacă produsurile sale sunt de forma: $x \rightarrow y, x, y \in (N \cup \Sigma)^*, |x|_N \geq 1, |x| \leq |y|$

Ce și în cazul de mai sus, dacă dorim ca $\lambda \in L(G)$, introducem producția $S \rightarrow \lambda$ astfel încât S să nu apară în membrul drept al niciunei producții.

= 13 =

Teorema Pentru orice gramatică monotone există o gramatică dependență de context echivalentă.

Reciproce acestei teoreme este evident căderea; deosebere o producție $xAy \rightarrow xzg$, $|z| \geq 1$, are evidență proprietatea $|x\cancel{Ay}| \leq |\cancel{xzg}|$.

Rezultă că gramaticile dependente de context și gramaticile monotone generează același familie de limbi, notată ca L_1 , care mai este numită în familia limbajelor dependente de context.

Limbajele din L_1 pot fi acceptate de automatele liniare mărginite, care sunt un caz particular de Mașini Turing.

GRAMATICI SI LIMBAJE DE TIPUL 2

Gramaticile independente de context se mai numesc în gramatici de tipul 2.

Familia limbajelor independente de context se mai notează ca L_2 .

Limbajele din L_2 sunt recunoscute de automatele pushdown.

GRAMATICI SI LIMBAJE DE TIPOUL 3

Gramaticile regulate se mai numesc și gramatici de tipul 3. Familia limbajelor generate de aceste gramatici (limbajele regulate) se mai notează și ca L_3 .

Limbajele din L_3 sunt recunoscute de automatele finite (deterministe sau neDeterministe, cu sau fără λ -transiții) și pot fi descrise de expresii regulate.

ALTE FAMILII DE LIMBAJE

LIMBAJELI LINIARE. Sunt limbajele generate de gramaticile independente de context liniare. Să notăm cu L_{lin} familia lor.

LIMBAJE RECURSIVE Sunt limbaje recursive enumerabile care sunt Turing decidabile (există o mașină Turing care pentru fiecare și în peste alfabetul unui limbaj decide dacă acel și este în limbaj sau nu).

Notăm cu L_{REC} aceste limbaje.

A nu se confunda cu notarea L_{RE} sau R.E. printre limbajele recursive enumerabile!

= 15 =

Terminia lui Chunesky. Există sursele
inclusivii stricte:

$$\mathcal{L}_3 \subsetneq \mathcal{L}_2 \subsetneq \mathcal{L}_1 \subsetneq \mathcal{L}_0$$

Exemplu

1) $G_1 = (\{S\}, \{a\}, S, \{S \rightarrow aS | a\})$ generează limbajul
 $L_1 = \{a^n | n \geq 1\}, L_1 \in \mathcal{L}_3$

2) $G_2 = (\{S, A, B\}, \{a, b\}, S, \{S \rightarrow AB, A \rightarrow aAb | \lambda, B \rightarrow aBb | \lambda\})$
 generează $L_2 = \{a^n b^n a^m b^m | n, m \geq 0\}, L_2 \in \mathcal{L}_0 - \mathcal{L}_1$

3) $G_3 = (\{S, B\}, \{a, b\}, S, \{S \rightarrow aBSa | aba, Ba \rightarrow aB, Bb \rightarrow bb\})$

generează $L_3 = \{a^n b^n a^n | n \geq 1\} \in \mathcal{L}_1 - \mathcal{L}_2$

O derivare în G_3 are forma:

$$\begin{aligned} S &\xrightarrow{n-1} (aB)^{n-1} Sa^{n-1} \Rightarrow (aB)^{n-1} aba a^n = \\ &= \underbrace{aBaB \dots aB}_{n-1 \text{ ori}} aba a^n \xrightarrow{*} a^n B^{n-1} ba a^n \Rightarrow \\ &\Rightarrow a^n B^{n-2} b^2 a^n \xrightarrow{*} a^n b^n a^n. \end{aligned}$$

4) Prin metoda diagonalelor se poate arăta
 că există $L_4 \in \mathcal{L}_0 - \mathcal{L}_1$

= 16 =

Rapinarea ierarhiei lui Chomsky -

Există următoarele incluzuni stricte

$$L_3 \subsetneq L_{\text{lin}} \subsetneq L_2 \subsetneq L_1 \subsetneq L_{\text{REC}} \subsetneq L_\omega$$

Exemplu

Gramatica $G_5 = (\{S\}, \{a, b\}, S, \{S \rightarrow aSb | ab\})$

$$\text{generează } L_5 = \{a^n b^n \mid n \geq 1\} \in L_{\text{lin}} - L_3$$

Se poate arăta că $L_2 \in L_2 - L_{\text{lin}}$

De asemenea, există limbi pe recursive care nu sunt dependente de context și există limbi pe recursive enumerabile care nu sunt recursive.