Limbaje Formale

Liviu P. DINU
Bucharest University, Faculty of Mathematics,
Academiei 14, RO-70109, Bucharest, Romania
E-mail: ldinu@funinf.cs.unibuc.ro
http://funinf.cs.unibuc.ro/$\sim$ ldinu

November 22, 2003

Contents

1 Preliminarii 2
1.1 Ierarhia Chomsky 3

2 Limbaje regulate 6
2.1 Gramatici regulate 6
2.2 Automate cu Stări Finite 7
2.2.1 Automate Finite Deterministe (AFD) 7
2.2.2 Automate Finite Nedeterministe (AFN) 9
2.3 Gramatici regulate şi Automate cu număr finit de stări 10
2.4 Proprietăţi de inchidere ale limbajelor regulate. Lema Bar-Hillel 11
2.5 Automatul minimal 11

3 Limbaje Independente de Context 13
3.1 Gramatici Independente de Context 13
3.2 Automate Pushdown Nedeterministe (APD) 14
3.3 Limbaje Independente de Context şi Automate Pushdown . . 16
3.4 Proprietăţi de inchidere 17

Chapter 1

Preliminarii

Această secţiune conţine noţiuni şi definiţii elementare despre alfabet, cuvinte, concatenare, monoid libe generat, lungimea cuvintelor, gramatici, limbaj, etc.

Un *alfabet* este o mulţime finită nevidă. Elementele unui alfabet Σ se numesc *litere*. Un *cuvânt* este o secvenţă finită de zero sau mai multe litere ale lui Σ; cuvântul fără nici o literă se numeşte *cuvânt vid* şi este notat cu λ.

Mulţimea tuturor cuvintelor peste Σ se notează cu Σ^*, în timp ce mulţimea tuturor cuvintelor nevide peste Σ se notează cu $\Sigma^+ = \Sigma^* - \{\lambda\}$. *Concatenarea* a două cuvinte u, v, notată uv, este obţinutaă prin juxtapunere, i.e., prin scrierea lui v după u. Mulţimea Σ^* este *monoidul liber* generat de Σ cu operaţia de concatenare. *Lungimea* unui cuvânt w, notată $|w|$, este dată de numărul literelor care apar în w; fiecare literă este contorizată ori de câte ori apare. Un *limbaj* peste un alfabet Σ este orice submulţime a lui Σ^*.

Definiţia 1. *O gramatică G este un sistem (T, N, S, P) unde T şi N sunt alfabete disjuncte (numite alfabetul terminalelor, respectiv neterminalelor), $S \in N$ este simbolul de start iar P este o mulţime finită şi nevidă a.i.*

$$P \subseteq (N \cup T)^* N (N \cup T)^*,$$

numită mulţimea producţiilor.

Notaţia 1. *Elementele lui P le notăm cu $u \to v$.*

Definiţia 2. *Fie G=(N,T,S,P) o gramatică. Definim relaţia $\Rightarrow_G \subseteq (N \cup T)^* \times (N \cup T)^*$ prin $w_1 \Rightarrow w_2$ (citim w_1 derivează direct in w_2) iff există x,*

y, u, $v \in (N \cup T)^*$ a.i. $w_1 = xuy$, $w_2 = xvy$ şi $u \to v \in P$. Când nu există pericol de confuzie renunţăm la indicele G din rela ctia definită mai sus.

Observaţia 1. *Relaţia de mai sus nu este neapărat reflexivă şi tranzitivă.*

Definiţia 3. *Închiderea reflexivă şi tranzitivă a relaţiei $\Rightarrow$ o notăm $\overset{*}{\Rightarrow}$. Avem $u \overset{*}{\Rightarrow} v$ dacă fie u=v, fie există $k \geq 1$, există $u = w_0, w_1, w_2, \ldots, w_k = v$ a.i. $w_i \Rightarrow w_{i+1}$, $i = 0, \ldots k-1$.*

Definiţia 4. *Fie G=(N,T,S,P) o gramatică. Limbajul*

$$L(G) = \{w \mid w \in T^*, S \overset{*}{\Rightarrow} w\}$$

se numeşte limbajul generat de gramatica G.

Exemplul 1. *Fie gramatica*

$$G = (\{S\}, (a, b), S, \{S \to aSb, S \to ab\}).$$

Limbajul generat de G este $L(G) = \{a^n b^n \mid n \geq 1\}$.

1.1 Ierarhia Chomsky

Impunerea unor restricţii asupra formei producţiilor unei gramatici (Chomsky, 1958) a dus la apariţia unor familii de limbaje care ocupă un loc central in teoria limbajelor formale.

Definiţia 5. *O gramatică $G = (N, T, S, P)$ se numeşte:*

1. *dependentă de context (context sensitive) sau de tipul 1, dacă fiecare producţie $u \to v$ a sa satisface condiţia $|u| \leq |v|$.*
2. *independentă de context (context free) sau de tipul 2, dacă fiecare producţie $u \to v$ a sa satisface condiţia $|u| = 1$, $v \neq \lambda$.*
3. *regulată sau de tipul 3, dacă fiecare producţie $u \to v$ a sa satisface condiţia $|u| = 1$, $v \in T^* \cup T^*N$, $v \neq \lambda$.*

Orice gramatică se numeşte de tipul 0.

Definiţia 6. *Orice limbaj generat de o gramatică de tipul 3 (2,1,0) se numeste limbaj regulat (independent de context, dependent de context, oarecare) sau limbaj de tipul 3 (2,1,0).*

Este evident că orice gramatică de tipul i este de tipul i-1, i=3,2,1.

Familia limbajelor generate de gramatici de tipul i (i=0,1,2,3) se notează cu $\mathcal{L}_i$ (i=0,1,2,3). Sunt evidente incluziunile următoare:

$$\mathcal{L}_3 \subseteq \mathcal{L}_2 \subseteq \mathcal{L}_1 \subseteq \mathcal{L}_0.$$

Se va arăta că aceste incluziuni sunt stricte.

Lema 1. *Fie G=(N,T,S,P) o gramatică de tipul i (i=3,2,1). Există o gramatică G_1 echivalentă cu G a.i. simbolul iniţial S_1 al lui G_1 să nu apară in nici unul din cuvintele aflate in membrul al doilea al producţiilor gramaticii G_1.*

Demonstraţie: (schiţă) Fie G=(N,T,S,P) o gramatică de tipul i (i=3,2,1) şi fie $S_1 \notin N \cup T$. Considerăm gramatica $G_1 = (N \cup S_1, T, S_1, P_1)$, unde $P_1 = P \cup \{S_1 \to u \mid S \to u \in P\}$. Se arată uşor că G şi G_1 sunt echivalente, i.e. dacă $w \in L(G)$ atunci $w \in L(G_1)$ şi reciproc.

Definiţia 7. *O gramatică G=(N,T,S,P) se numeşte recursivă dacă pentru orice cuvânt $w \in T^+$ există un algoritm pentru a decide care din alternativele $w \in L(G)$ sau $w \notin L(G)$ are loc.*

Teorema 1. *Gramaticile dependente de context sunt recursive*

Demonstraţie: Fie G=(N,T,S,P) o gramatică de tipul 1 şi $w \in T^+$. Notăm $n = |w|$ şi definim recursiv mulţimile:

- $U_0 = \{S\}$
- $U_{m+1} = U_m \cup \{v \mid v \in (N \cup T)^+, \exists u[u \in U_m \ a.i. \ u \Rightarrow v \ si \ |v| \leq n\}$

Se demonstrează uşor că au loc următoarele proprietăţi:

1. $U_m = \{v \mid v \in (N \cup T)^+, \ S \overset{\leq m}{\Rightarrow} |v| \leq n\}$
2. $U_0 \subseteq U_1 \subseteq \ldots \subseteq U_m \subseteq \ldots \subseteq \bigcup_{t=1}^{t=n} (N \cup T)^t$
3. există $k \in \mathbf{N}$ a.i. $U_k = U_k + 1$. Daca $U_k = U_{k+1}$ atunci $U_k = U_{k+i}$, pentru orice i=1,2,...
4. Fie k_0 cel mai mic număr natural a.i.$U_k = U_{k+1}$. Atunci

$$w \in L(G) \; iff \; w \in U_{k_0}.$$

Definiţia 8. *O producţie de forma $X \rightarrow Y$, X şi Y neterminale, se numeşte redenumire.*

Propoziţia 1. *Fie G=(N,T,S,P) o gramatică de tipul 2 sau 3. Există o gramatică G_1 echivalentă cu G şi fără redenumiri.*

Demonstraţie: Fie G=(N,T,S,P) o gramatică de tipul 2 sau 3. Fie

$$P_1 = \{A \rightarrow u \mid u \notin N,\ A \rightarrow u \in P\}$$
$$P_2 = \{A \rightarrow u \mid A \in N,\ \exists B[B \in N \text{ a.i. } A \overset{+}{\Rightarrow}_G B,\ B \rightarrow u \in P_1]\}$$

Producţiile din P_2 nu sunt redenumiri. Fie $P' = P_1 \cup P_2$. Gramatica $G_1 = (N, T, S, P')$ este fără redenumiri şi se arată uşor că este echivalentă cu G.

Chapter 2

Limbaje regulate

2.1 Gramatici regulate

Reamintim că o gramatică regulată sau de tipul 3, este o gramatică in care fiecare producţie $u \to v$ a sa satisface condiţia $|u| = 1$, $v \in T^* \cup T^*N$, $v \neq \lambda$.

Propoziţia 2. *Pentru orice gramatică regulată G=(N,T,S,P) există o gramatică $G_1 = (N_1, T, S_1, P_1)$ exchivalentă cu ea şi având proprietatea că fiecare producaţie $u \to v \in P_1$ a sa satisface condiţiile $u \in N_1$, $v \in T \cup TN_1$.*

Demonstraţie: Cf. Lemei anterioare, există o gramatică echivalentă G' cu G şi fără redenumiri. Producţiile ei sunt fie de forma $A \to a_1a_2 \dots a_k$, $k \geq 1$ cu $a_i \in T$, i=1,2,...k, fie de forma $A \to a_1a_2 \dots a_kB$, $k \geq 1$ cu $a_i \in T$, i=1,2,...k, $B \in N$. În cazul k=1 aceste producţii sunt de forma $A \to a$, $a \in T$ şi le adăugăm noii gramatici G_1. Când $k \geq 2$, pentru fiecare producţie adăugăm neterminalele noi $A_1, A_2, \dots, A_{k-1}$ şi producţiile:

$$\begin{gathered} A \to a_1A_1 \\ A_1 \to a_2A_2 \\ \cdots\cdots\cdots \\ A_{k-1} \to a_k \end{gathered}$$

in locul producţiei $A \to a_1a_2 \dots a_k$, iar in locul producţiei $A \to a_1a_2 \dots a_kB$ adăugăm producţiile:

$$\begin{gathered} A \to a_1A_1 \\ A_1 \to a_2A_2 \end{gathered}$$

$$\cdots\cdots\cdots$$
$$A_{k-1} \rightarrow a_k B$$

Gramatica G_1 va avea aceleaşi terminale ca şi G', neterminalele vor fi cele vechi la care le adăugăm pe cele nou introduse, simbolul de start va fi acelaşi, iar producţiile vor fi cele pe care le-am introdus cf. procedurilor de mai sus. Nu este greu de arătat că G_1 şi G sunt echivalente.

Propoziţia anterioară ne permite construirea unui algoritm eficient pentru a decide dacă un cuvânt dat aparttine sau nu unui limbaj regulat.

Observaţia 2. *Fie o gramatică regulată G=(N,T,S,P) cu producţiile in forma canonică şi un cuvânt $w = w_1 w_2 \ldots w_k$, $w_i \in T$, i=1,2,...,k. Putem constata care din alternativele $w \in L(G)$ sau $w \notin L(G)$ are loc alcătuind recursiv mulţimile:*

- $U_0 = \{X \mid X \in N \text{ a.i. } X \rightarrow w_k \in P\}$
- $U_m = \{X \mid X \in N,\ \exists Y[Y \in U_{m-1} \text{ a.i. } X \rightarrow w_{k-m} Y \in P]\}$, *m=1,2,. . . , k-1.*

Avem $w \in L(G)$ iff $S \in U_{k-1}$.

2.2 Automate cu Stări Finite

2.2.1 Automate Finite Deterministe (AFD)

Definiţia 9. *Se numeşte automat finit determinist un quintuplu* $(\Sigma, Q, q_0, \delta, F)$, *unde:*

1. *Σ este un alfabet numit alfabetul de intrare*
2. *Q este o mulţime finită nevudă numită mulţimea stărilor interne*
3. *$q_0 \in Q$ este starea iniţială a automatului*
4. *$\delta : Q \times \Sigma \rightarrow Q$ se numeşte funcţia de tranziţie a automatului*
5. *$F \subseteq Q$ este mulţimea stărilor finale ale automatului*

Extindem funcţia de tranziţie la cuvinte in felul următor:

Definiţia 10. *$\overline{\delta} : Q \times \Sigma^* \rightarrow Q$ este definită astfel:*

- $\overline{\delta}(q,\lambda) = \lambda$
- $\overline{\delta}(q,wx) = \delta(\overline{\delta}(q,w),x)$,

pentru orice $w \in \Sigma^*$, *orice* $x \in \Sigma$ *şi orice* $q \in Q$.

Cu alte cuvinte, $\overline{\delta}(q,w)$ este starea in care ajunge automatul plecând din starea q şi primind la intrare cuvântul w.

Observaţia 3. *Funcţia* $\overline{\delta}$ *extinde funcţia* δ, *deoarece dacă* $(q,a) \in Q \times \Sigma$, *avem*

$$\overline{\delta}(q,a) = \overline{\delta}(q,\lambda a) = \delta(\overline{\delta}(q,\lambda),a) = \delta(q,a)$$

Vom nota extensia $\overline{\delta}$ tot cu δ pentru uşurinţa scrierii.

Observaţia 4.

Definiţia 11. *Fie* $A = (\Sigma, Q, q_0, \delta, F)$ *un AFD. Limbajul acceptat de automatul A este:*

$$L(A) = \{w \mid w \in \Sigma^*,\ \delta(q_0,w) \in F\}.$$

Nu este greu de demonstrat următoarele două leme:

Lema 2. *Fie A un AFD. Fiind dată starea* $\delta(q,w) = s$, *cu* $w \neq \lambda$, $w = w_1w_2\dots w_n$, $w_i \in \Sigma$, $i = 1,2,...n$, *există stările* $q_1, q_2, \dots q_{n+1}$ *a.i.* $q_1 = q, q_{n+1} = s, q_{i+1} = \delta(q_i,w_i)$, $i = 1,2,...,n$.

Lema 3. *Fie A un AFD. Fiind date stările* $q_1, q_2, \dots q_{n+1}$ *a.i.* $q_{i+1} = \delta(q_i,w_i)$, $i = 1,2,...,n$, *atunci* $q_{n+1} = \delta(q_1,w)$, *cu* $w = w_1w_2\dots w_n$, $w_i \in \Sigma$, $i = 1,2,...n$

Corolar 2.2.1. *Într-un AFD, dacă w=uv, atunci* $\delta(q,w) = \delta(\delta(q,u),v)$.

Definiţia 12. *Mulţimea stărilor accesibile ale unui AFD* $A = (\Sigma, Q, q_0, \delta, F)$ *este mulţimea*

$$Q_a = \{q \mid q \in Q,\ ,\exists w[w \in \Sigma^* \ a.i.\ \delta(q_0,w) = q]\}$$

Cu alte cuvinte, stările accesibile ale unui automat sunt acele stări in care se poate ajunge pornind din starea iniţială şi primind la intrare un cuvânt oarecare w.

Stările accesibile pot fi calculate cu următorul algoritm:

- $U_0 = \{q_0\}$
- $U_{m+1} = U_m \cup \{q \mid q \in Q,\ \exists a \exists s[a \in \Sigma,\ s \in U_m,\ a.i.\ \delta(s,a) = q]\}$

Cel mai mic $i \in \mathbf{N}$ pentru care $U_i = U_{i+1}$ ne permite determinarea stărilor accesibile: $Q_a = U_i$.

Definiţia 13. *Mulţimea stărilor observabile ale unui AFD* $A = (\Sigma, Q, q_0, \delta, F)$ *este mulţimea*

$$Q_o = \{q \mid q \in Q,\ ,\exists w[w \in \Sigma^*\ a.i.\ \delta(q,w) \in F]\}$$

2.2.2 Automate Finite Nedeterministe (AFN)

Definiţia 14. *Un AFN este este un quintuplu* $(\Sigma, Q, Q_0, \delta, F)$*, unde:*

1. Σ *este un alfabet numit alfabetul de intrare*
2. Q *este o mulţime finită nevudă numită mulţimea stărilor interne*
3. $Q_0 \subseteq Q$ *este o mulţime nevidă, numită mulţimea stărilor iniţiale ale automatului*
4. $\delta : Q \times \Sigma \to 2^Q$ *se numeşte funcţia de tranziţie a automatului*
5. $F \subseteq Q$ *este mulţimea stărilor finale ale automatului*

Definiţia 15. *Fie* $A = (\Sigma, Q, Q_0, \delta, F)$ *un AFN. Limbajul acceptat de A este format din toate cuvintele* $w = w_1 w_2 \ldots w_n$ $(w_i \in \Sigma,\ i = 1, 2, ..., n)$ *pentru care există* $q_1, q_2, \ldots, q_{n+1}$ *cu* $q_1 \in Q_0,\ q_{n+1} \in F$ *şi* $q_{i+1} \in \delta(q_i, w_i)$*, i=1,2,...,n.*

Este evident că orice AFD poate fi privit ca un AFN, prin urmare avem următoarea teoremă:

Teorema 2. *Limbajul reprezentabil intr-un AFD este reprezentabil intr-un AFN*

Vom arăta că şi reciproca este adevărată.

Teorema 3. *Limbajul reprezentabil intr-un AFN este reprezentabil intr-un AFD.*

Demonstraţie: (construcţie) Fie L un limbaj reprezentat in AFN-ul $A = (\Sigma, Q, Q_0, \delta, F)$. Considerăm AFD $A_1 = (\Sigma, 2^Q, Q_0, \delta', F_1)$, unde funcţia de tranziţie este definită astfel:

- $\delta'(P, a) = \emptyset$, dacă $P = \emptyset$
- $\delta'(P, a) = \bigcup_{q \in P} \delta(q, a)$, dacă $P \neq \emptyset$,

iar mulţimea stărilor finale este: $F_1 = \{S \mid S \subseteq Q, S \cap F \neq \emptyset\}$.

2.3 Gramatici regulate şi Automate cu număr finit de stări

Teorema 4. *Orice limbaj regulat este reprezentabil intr-un AFN (AFD).*

Demonstraţie: (construcţie) Fie L un limbaj regulat; există aąsadar o gramatică regulată $G = (N, T, S, P)$ care il generează a.i. simbolul de start al gramaticii nu apare in membrul drept al nici unei producţii şi ale cărei producţii sunt de forma $A \to a$ sau $A \to aB$, cu $A, B \in N$ şi $a \in T$.

Fie $X \notin T \cup N$ şi automatul finit determinist $A = (T, N \cup \{X\}, \{S\}, \delta, F_1)$, unde $\delta : (N \cup \{X\}) \times T \to 2^{N \cup \{X\}}$ este dată de:

- $\delta(Y, a) = \emptyset$ dacă $Y = X$
- $\delta(Y, a) = \{A \mid Y \to aA\}$ dacă $Y \neq X$ şi $Y \to a \notin P$
- $\delta(Y, a) = \{A \mid Y \to aA\} \cup \{X\}$ dacă $Y \neq X$ şi $Y \to a \in P$

Teorema 5. *Orice limbaj reprezentabil intr-un AFD (AFN) este regulat.*

Demonstraţie: (construcţie) Fie AFD $A = (\Sigma, Q, q_0, \delta, F)$ care acceptă limbajul L. Construim gramatica $G = (N, T, S, P)$, unde:

1. $T = \Sigma$;
2. N=Q;
3. $S = q_0$;
4. Producţiile sunt definite astfel: 1) $A \to a \in P$ iff $\delta(A, a) \in F$; 2) $A \to aB$ iff $\delta(A, a) = B$, $A, B \in Q$, $a \in T$.

Se arată uşor că limbajul generat de gramatica G este identic cu L.

2.4 Proprietăţi de inchidere ale limbajelor regulate. Lema Bar-Hillel

Teorema 6. *Limbajele regulate sunt inchise la următoarele operaţii:*

1. *reuniune: dacă* $L_1,\ L_2 \in \mathcal{L}_3$ *atunci* $L_1 \cup L_2 \in \mathcal{L}_3$
2. *intersecţie: dacă* $L_1,\ L_2 \in \mathcal{L}_3$ *atunci* $L_1 \cap L_2 \in \mathcal{L}_3$
3. *concatenare: dacă* $L_1,\ L_2 \in \mathcal{L}_3$ *atunci* $L_1 L_2 \in \mathcal{L}_3$
4. *complementară: dacă* $L \in \mathcal{L}_3$ *atunci* $\mathcal{C}L \in \mathcal{L}_3$
5. *inchiderea Kleene: dacă* $L \in \mathcal{L}_3$ *atunci* $L^* = \bigcup_{i \geq 0} L^i \in \mathcal{L}_3$*;*
6. *dacă* $L \in \mathcal{L}_3$ *atunci* $Sub(L) = \{w \mid \exists x \in \Sigma^*, \exists y \in \Sigma^* \text{ a.i. } xwy \in L\}$ *este un limbaj regulat.*

2.5 Automatul minimal

Ne punem problema determinării unui automat cu un număr minim de stări intermediare care să accepte un limbaj regulat dat.

Propoziţia 3. *Fie* $A = (\Sigma, Q, q_0, \delta, F)$ *un AFD.*

1. *Pentru orice număr natural* k *se defineşte relaţia* $\overset{k}{\equiv}$*pe* Q *după cum urmează:*

$$q_1 \overset{k}{\equiv} q_2 \text{ iff } \forall w [w \in \bigcup_{i=0}^{k} \Sigma^i \text{ avem } \delta(q_1, w) \in F \Leftrightarrow \delta(q_2, w) \in F]$$

2. *Pe* Q *definim relaţia* $\equiv$ *astfel:*

$$q_1 \equiv q_2 \text{ iff } \forall k [k \in \mathbf{N},\ q_1 \overset{k}{\equiv} q_2].$$

Relaţiile $\overset{k}{\equiv}$ *şi* $\equiv$ *sunt relaţii de echivalenţă.*

Propoziţia 4. *Au loc următoarele proprietăţi:*

1. $q_1 \overset{0}{\equiv} q_2$ *iff* q_1 *şi* q_2 *sunt simultan in* F *sau in* Q*-*F*.*

2. dacă $k \geq 1$ atunci $q_1 \overset{k}{\equiv} q_2$ iff $\delta(q_1, a) \overset{k-1}{\equiv} \delta(q_2, a)$, $\forall a \in \Sigma \cup \{\lambda\}$.

Propoziţia 5. *Relaţiile de echivalenţă $\overset{k}{\equiv}$ satisfac proprietăţile:*

1. $\equiv \subseteq \ldots \subseteq \overset{k}{\equiv} \subseteq \overset{k-1}{\equiv} \subseteq \ldots \subseteq \overset{0}{\equiv}$

2. există $k_0 \in \mathbf{N}$ a.i. $\overset{k_0}{\equiv} = \overset{k_0+i}{\equiv}$, $\forall i \geq 1$

3. $\equiv = \overset{k_0}{\equiv}$.

Propoziţia 6. *Notăm $Q_1 = Q/_{\equiv}$. Definim funcţia*

$$\delta_1 : Q_1 \times \Sigma \to \Sigma \text{ prin } \delta_1([q], a) = [\delta(q, a)] \text{ penrtu orice } a \in \Sigma,$$

unde prin [q] am notat clasa de echivalenţă a lui q in raport cu relaţia $\equiv$. Să se arate că δ_1 este bine definită.

Propoziţia 7. *Să se arate că $\delta_1([q], w) = [\delta(q, w)]$, pentru orice $w \in \Sigma^*$.*

Teorema 7. *Fie $A = (\Sigma, Q, \delta, q_0, F)$ un AFD fără stări inaccesibile. Automatul cu număr minim de stări care acceptă acelaşi limbaj ca şi A este $A_{min} = (\Sigma, Q_1, \delta_1, [q_0], F/_{\equiv})$, unde elementele sale sunt definite cf. propoziţiilor anterioare.*

Lema 4. *(Lema de pompare sau lema Bar-Hillel) Dacă L este un limbaj regulat, atunci există $k \in \mathbf{N}^*$ a.i. oricare ar fi $w \in L$, $|w| > k$, are o descompunere de forma $w = xyz$, cu $x, y, z \in \Sigma^*$, $0 < |y| \leq k$ şi $xy^iz \in L$, pentru orice $i \geq 0$.*

Demonstraţie: Fie $A = (\Sigma, Q, \delta, q_0, F)$ un AFD minimal cu k stări a.i. să accepte limbajul L.

Dacă $w \in L$ şi $|w| > k$ scriem $w = w_1 w_2 \ldots w_n$ şi considerăm secvenţa de stări:

$$\delta(q_0, w_1),\ \delta(q_0, w_1 w_2) \ldots \delta(q_0, w_1 w_2 \ldots w_{k+1})$$

În secvenţa de mai sus există două stări egale şi de aici demonstraţia decurge uşor.

Chapter 3

Limbaje Independente de Context

3.1 Gramatici Independente de Context

Propoziţia 8. *Orice limbaj independent de context poate fi generat de o gramatică G=(N,T,S,P) de tipul 2 ale cărei producţii sunt de forma* $X \rightarrow A_1 A_2 \dots A_k$ *sau* $X \rightarrow a$*, unde* $k > 1$*,* $X, A_1, A_2, \dots, A_k \in N$ *şi* $a \in T$*.*

Demonstraţie: Există o gramatică echivalentă G' cu G şi fără redenumiri care il genrează pe L. Producţiile ei sunt fie de forma $A \rightarrow A_1 A_2 \dots A_k$, $k \geq 2$ cu $A_i \in T \cup N$, i=1,2,...k, fie de forma $A \rightarrow a$, cu $a \in T$. In cazul al doilea aceste producţii sunt de forma dorită şi le lăsăm neschimbate. În primul caz, pentru fiecare terminal A_i adăugăm un nou neterminal B_i distinct de toate celelalte şi inlocuim in producţie pe A_i cu B_i, adăugând producţia $B_i \rightarrow A_i$.

Propoziţia 9. *(Forma normală Chomsky) Orice limbaj independent de context poate fi generat de o gramatică G=(N,T,S,P) de tipul 2 ale cărei producţii sunt de forma* $X \rightarrow A_1 A_2$ *sau* $X \rightarrow a$*, unde* $X, A_1, A_2 \in N$ *şi* $a \in T$*.*

Demonstraţie: (construcţie) Din propoziţia anterioară am văzut că orice limbaj independent de context poate fi generat de o gramatică de tipul 2 fără redenumiri şi ale cărei producţii sunt de forma: $A \rightarrow A_1 A_2 \dots A_k$, $k \geq 2$ cu $A_i \in N$, i=1,2,...k, sau de forma $A \rightarrow a$, cu $a \in T$. Pentru fiecare producţie de forma $A \rightarrow A_1 A_2 \dots A_k$, $k \geq 2$ vom introduce k-2 neterminale noi $B_1, B_2, \dots, B_{k-2}$ şi producţiile:

$$\begin{gathered} A \to A_1B_1 \\ B_1 \to A_2B_2 \\ \cdots\cdots\cdots \\ B_{k-2} \to A_{k-1}A_k \end{gathered}$$

Teorema 8. *Fie G=(N,T,S,P) o gramatică independentă de context in FNC. Dacă derivarea* $S \stackrel{*}{\Rightarrow}$ *are proprietatea că cel mai lung lanţ de la rădăcină la vârfurile terminale in arborele de derivare asociat ei are k noduri, atunci* $|w| \leq 2^{k-1}$.

Teorema 9. *(Lema de pompare) Pentru orice limbaj independent de context L există numerele naturale p şi q a.i. orice cuvânt* $w \in L$ *cu* $|w| > p$ *poate fi scris sub forma w=xyzuv, unde:*

1. $|yzu| \leq q$;
2. $yu \neq \lambda$
3. $xy^izu^iv \in L$, *pentru orice* $i \geq 0$.

3.2 Automate Pushdown Nedeterministe (APD)

Definiţia 16. *Un APD este un sistem* $(K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ *unde:*

1. *K este o mulţime nevidă, finită, repezentând mulţimea stărilor*
2. Σ *este un alfabet, numit alfabetul de intrare*
3. Γ *este un alfabet, numit alfabetul stivei*
4. $\delta : K \times (\Sigma \cup \{\lambda\}) \times \Gamma \to 2^{K \times \Gamma^*}$
5. $q_0 \in K$ *este starea iniţială*
6. $Z_0 \in \Gamma$ *este simbolul de start al stivei*
7. $F \subseteq K$ *este mulţimea stărilor finale.*

Definiţia 17. *Se numeşte configuraţie a unui automat pushdown P un triplet* $(q, w, u) \in K \times \Sigma^* \times \Gamma^*$, *unde:*

1. $q \in K$ *este starea in care se află automatul*

2. $w \in \Sigma^*$ *este partea necitită din cuvântul aflat pe banda de intrare.*

3. u *reprezintă conţinutul stivei*

Pe mulţimea configuraţiilor definim o relaţie binară notată $\vdash$ definită astfel:

Definiţia 18. *Configuraţia* (q, aw, Zu) *se află in relaţia* $\vdash$ *cu configuraţia* (s, w, vu) *şi scriem*

$$(q, aw, Zu) \vdash (s, w, vu)$$

dacă $(s, v) \in \delta(q, a, Z)$*, unde* $q, s \in K$*,* $a \in \Sigma \cup \{\lambda\}$*,* $Z \in \Gamma$*,* $w \in \Sigma^*$*,* $u, v \in \Gamma^*$*.*

Într-un mod analog gramaticilor regulate definim inchiderea reflexivă şi tranzitivă a relaţiei $\vdash$.

Definiţia 19. *Cuvântul* $w \in \Sigma^*$ *este acceptat de automatul pushdown P dacă există* $q \in F$*,* $u \in \Gamma^*$ *a.i.* $(q_0, w, Z_0) \overset{*}{\vdash} (q, \lambda, u)$

Definiţia 20. *Fie* $P = (K, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset)$ *un APD. Spunem că un cuvânt* $w \in \Sigma^*$ *este acceptat de P cu memoria pushdown vidă dacă există* $q \in K$ *a.i.* $(q_0, w, Z_0) \overset{*}{\vdash} (q, \lambda, \lambda)$

Notaţia 2. *Vom nota cu* $L_\lambda(P)$ *mulţimea tuturor cuvintelor acceptate de P cu memoria vidă.*

Teorema 10. *Fie L(P) limbajul acceptat de un APD* $P = (K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$*. Există un APD* P_1 *a.i.* $L(P) = L_\lambda(P_1)$*.*

Demonstraţie: Fie q_λ, q_0' două elemente distincte ce nu apar in K şi fie X, $X \notin \Gamma$. Construim APD cu memorie vidă $P_1 = (K \cup \{q_\lambda, q_0'\}, \Sigma, \Gamma \cup \{X\}, \delta_1, q_0', X, \emptyset)$, unde δ_1 o definim astfel:

1. $\delta_1(q, a, Z) = \delta(q, a, Z)$, dacă $q \in K, a \in \Sigma,\ Z \in \Gamma$
   $\delta_1(q, \lambda, Z) = \delta(q, \lambda, Z)$, dacă $q \in K - F,\ Z \in \Gamma$
   $\delta_1(q, \lambda, Z) = \delta(q, \lambda, Z) \cup \{(q_\lambda, \lambda)\}$, dacă $q \in F,\ Z \in \Gamma$

2. $\delta_1(q, \lambda, X) = \{(q_\lambda, \lambda)\}$, dacă $q \in F$

3. $\delta_1(q_0', \lambda, X) = \{(q_0, Z_0X)\}$

4. $\delta_1(q_\lambda, \lambda, Z) = \{(q_\lambda, \lambda)\}$ dacă $Z \in \Gamma \cup \{X\}$

5. $\emptyset$ in celelalte cazuri

Teorema 11. *Fie L(P) limbajul acceptat de un APD* $P = (K, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset)$. *Există un APD* P_1 *a.i.* $L(P_1) = L_\lambda(P)$.

Demonstraţie: Fie $P_1 = (K \cup \{q_0', q_f\}, \Sigma, \Gamma \cup \{X\}, \delta_1, q_0', X, q_f)$, $q_0' \neq q_f \notin K$, $X \notin \Gamma$,iar δ_1 o definim astfel:

1. $\delta_1(q, a, Z) = \delta(q, a, Z)$, dacă $q \in K, a \in \Sigma \cup \{\lambda\},\ Z \in \Gamma$

2. $\delta_1(q, \lambda, X) = \{(q_f, \lambda)\}$, dacă $q \in K$

3. $\delta_1(q_0', \lambda, X) = \{(q_0, Z_0X)\}$

4. $\emptyset$ in celelalte cazuri

3.3 Limbaje Independente de Context şi Automate Pushdown

Teorema 12. *Fie L un limbaj independent de context. Există un APD P a.i.* $L_\lambda(P) = L$.

Demonstraţie: Dacă $L \in \mathcal{L}_2$, există o gramatică independentă de context G=(N,T,S,P) a.i. L=L(G). Construim automatul pushdown $P = (\{q\}, T, T \cup N, \delta, q, S, \emptyset)$, unde δ este dată de:

1. $\delta(q, \lambda, A) = \{(q, u) \mid A \to u \in P\}$

2. $\delta(q, a, a) = \{(q, \lambda)\}$, pentru $a \in T$

3. $\emptyset$ in celelalte cazuri.

Teorema 13. *Fie* $P = (K, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset)$ *un APD cu memoria vidă. Atunci limbajul* $L_\lambda(P)$ *este independent de context.*

Demonstraţie: Vom construi o gramatică independentă de context care să genereze $L_\lambda(P)$.

Fie G=(N,T,S,P), unde:

- $N = K \times \Gamma \times K \cup \{S\}$, unde $S \notin K \times \Gamma \times K$; fiecare neterminal (q,Z,r) il notăm [qZr].

- $T = \Sigma$

- Producţiile sunt definite după cum urmează:

  1. $S \to [q_0 Z_0 q]$, pentru orice $q \in K$;
  2. $[qZr] \to a$ pentru orice $a \in \Sigma \cup \{\lambda\}$ a.i. $(r, \lambda) \in \delta(q, a, Z)$
  3. $[qZs_k] \to a[ru_1s_1][s_1u_2s_2] \ldots [s_{k-1}u_ks_k]$ pentru orice insiruire de $k \geq 1$ stări $s_1, \ldots, s_k$ din K şi orice $a \in \Sigma \cup \{\lambda\}$ pentru care $(r, u_1u_2 \ldots u_k) \in \delta(q, a, Z)$, $u_i \in \Gamma$, i=1,2,...k.

3.4 Proprietăţi de inchidere

Teorema 14. *Limbajele independente de context sunt inchise la reuniune, concatenare, inchiderea Kleene, substituţii.*

Propoziţia 10. *Limbajele independente de context nu sunt inchise la intersecţie şi la complementară.*

Demonstraţie:

- $\{a^i b^i c^j \mid i, j, k \geq 1\} \cap \{a^i b^j c^j \mid i, j, k \geq 1\} = \{a^j b^j c^j \mid j \geq 1\} \notin \mathcal{L}_2$, deşi fiecare din cele două limbaje este independent de context.

- $\mathcal{C}\{a^j b^j c^j \mid j \geq 1\} \in \mathcal{L}_2$

Propoziţia 11. *Familia limbajelor independente de context este inchisă la intersecţia cu limbaje regulate.*

Bibliography

[1] . Atanasiu, A. Mateescu. *Limbaje Formale. Culegere de Probleme*, Ed. Univ. Bucureşti, 1990

[2] H. Georgescu, C. Popovici, S. Rudeanu. *Bazele Informaticii*, vol. II, Ed. Univ. Bucureşti, 1991, pg.1-90

[3] . Salomaa. *Formal Languages*, 1973

[4] . Salomaa, Rozenberg (eds.) *Handbook of Formal Languages*, Springer, 1997