Limbaje Formale și Compilatoare (LFC)
- Curs -

Ș.l.dr.ing. Octavian MACHIDON

Universitatea
Transilvania
din Brașov

Astăzi

- Limbaje formale
- Mecanisme de generare a limbajelor: Gramatici
- Ierarhia lui Chomsky
- Limbaje și gramatici de tip 3 (regulate)
- Proprietăți de închidere pentru familia de limbaje regulate

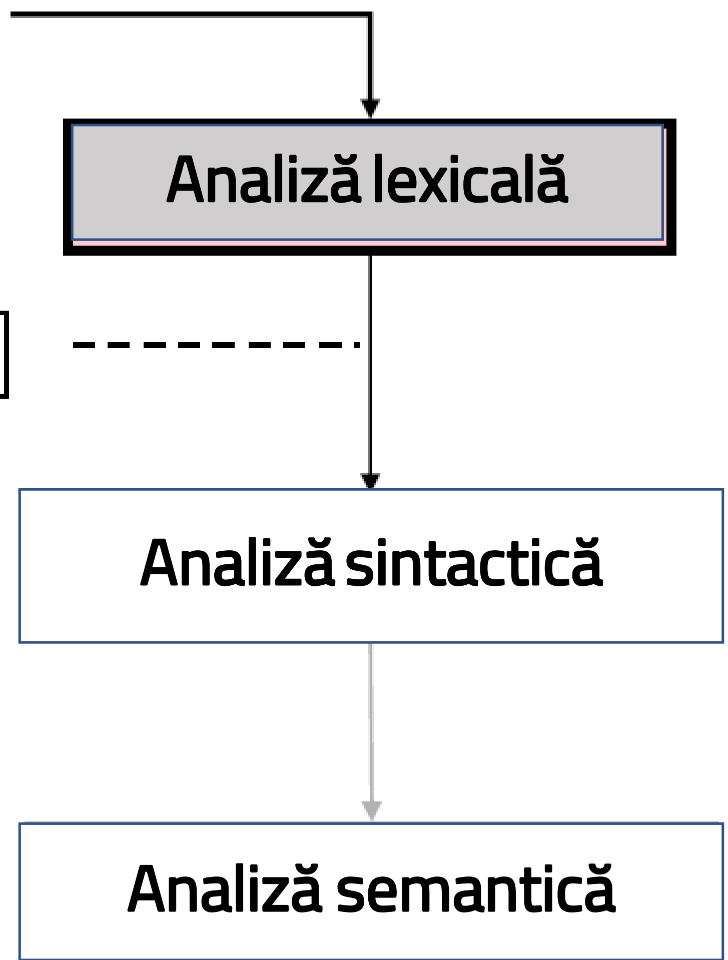
Primul pas al compilării: Analiza lexicală

- Cod sursă (șir de caractere)

```
if (b == 0) a = b;
```

- Șir de atomi lexicali

if	(	b	==	0	)	a	=	b	;



Atomii lexicali

- Identificatori: x   y11   elsen   _i00
- Cuvinte cheie: if else   while   break
- Constante:
  - Integer: 2   1000   -500   5L   0x777
  - Floating-point: 2.0   0.00020   .02   1.   1e5   0.e-10
  - String: “x”   ”He said, \”Are you?\”\n”
  - Character: ’c’   ’\000’
- Simboluri: +   *   {   }   ++   <   <<   [   ]   >=
- Spații (de obicei recunoscute și ignorate):
  - Comentarii: /** don’t change this **/
  - Spațiu: <space>
  - Caractere de formatare: <newline> <return>

Analizorul lexical: „lexer”

- Poate fi implementat „ad-hoc”...?
  - Exemplu: implementare care citește atomii de tip identificatori
  - Probleme:
    - Cum se începe?
    - Ce se face cu următorul caracter?
    - Cum se recunosc cuvintele
    - Cum se poate reduce complexitatea generată de concatenări repetate?
  - E nevoie de „look-ahead”:
    - Probleme rămân: cum știm ce atom avem la citirea primului caracter?
- E necesară o abordare principială:
  - Teoria limbajelor

```
Token readIdentifier( ) {
  String id = "";
  while (true) {
    char c = input.read();
    if (!identifierChar(c))
          return new Token(ID, id, lineNumber);
    id = id + String(c);
  }
}
```

```
char next;
...
while (identifierChar(next)) {
  id = id + String(next);
  next = input.read ();
}
```

Alfabet, cuvânt, mulțime de cuvinte

- Alfabet: V - o mulțime finită (elementele lui V = simboluri)
- Cuvânt: șir finit de simboluri
- cuvântul nul este notat cu ε sau λ.
- Lungimea unui cuvânt u : numarul simbolurilor sale. Notație: $|u|$.
- $|\varepsilon| = 0$
- V – mulțimea tuturor cuvintelor peste alfabetul V, inclusiv ε.
- {0, 1} = {ε, 0, 1, 00, 01, 10, 11, 000, 001, . . .}
- V+ - mulțimea tuturor cuvintelor nenule peste alfabetul V
- {0, 1}+ = {0, 1, 00, 01, 10, 11, 000, 001, . . .}

Operații pe cuvinte

- **Concatenarea** a doua cuvinte x, y este cuvântul $x \cdot y$ obținut din simbolurile lui x, în ordinea în care apar, urmate de cele ale lui y de asemenea în ordinea în care apar:

  $x = 0100$, $y = 100$, $x \cdot y = 0100100$

  $x = 000$, $y = \varepsilon$, $x \cdot y = 000$

- Concatenarea este asociativă

Alfabet și limbaj

- Fie V un alfabet. O submulțime $L \subseteq V$ este un limbaj (formal) peste alfabetul V (sau V-limbaj) dacă L are o descriere (matematică) finită.
- O descriere poate fi:
  - neformală (în limbaj natural):
    - multimea cuvintelor peste alfabetul {0, 1} care contin un numar par de 0.
    - $L = \{x \in V^+ : |x|$ *este par*$\}$.
    - $\{a^n b^n | n \in N\}$.
    - $\{w \in \{0, 1\}^* | w$ *se termina in* $00\}$.
- formală (descriere matematică):
    - o descriere inductivă a cuvintelor
    - o descriere generativă a cuvintelor (gramatică generativă)
    - o descriere a unei metode de recunoaștere a cuvintelor din limbaj (automat finit, automat pushdown, etc.)

Operații cu limbaje

- Operațiile cu mulțimi (reuniune, intersecție etc..)
- Produs de limbaje: $L_1 \cdot L_2 = \{u \cdot v | u \in L_1, v \in L_2\}$
- Iterația (produsul Kleene): $L^* = \bigcup_{n \geq 0} L^n$, unde:
  - $L_0 = \{\varepsilon\}$
  - $L_{n+1} = L_n \cdot L$
- $L^R = \{w^R | w \in L\}$; dacă $w = a_1 a_2 \ldots a_n$, atunci $w^R = a_n \ldots a_2 a_1$

Gramatici

- O gramatică este un sistem $G = (N, T, S, P)$, unde:
- N și T sunt două alfabete disjuncte:
  - N este mulțimea simbolurilor neterminale
  - T este mulțimea simbolurilor terminale
- $S \in N$ este simbolul de start (simbolul neterminal inițial)
- P este o mulțime finită de reguli (producții) de forma $x \rightarrow y$, unde $x, y \in (N \cup T)^*$ și x conține cel puțin un simbol neterminal.

Derivare

- Fie G = (N, T, S, P) o gramatică și u, v ∈ $(N \cup T)^*$.
- Spunem că *v este derivat direct (într-un pas) din u* prin aplicarea regulii $x \to y$, și notăm $u \Rightarrow v$, dacă ∃p, q ∈ $(N \cup T)^*$ astfel încât u =pxq și v =pyq.
- Daca $u_1 \Rightarrow u_2 \ldots \Rightarrow u_n$, $n > 1$, spunem ca u_n este derivat din u_1 în *G* și notăm $u_1 \Rightarrow^+ u_n$.
- Scriem $u \Rightarrow^* v$ dacă $u \Rightarrow^+ v$ sau $u = v$.

Limbaj generat

- Limbajul generat de gramatica G este:

  $L(G) = \{w \in T^* | S \Rightarrow^+ w\}$

- Două gramatici G_1 și G_2 sunt echivalente dacă:

  $L(G_1) = L(G_2)$.

Exemple (1)

- $L = \{a^n b^{2n} | n \geq 1\}$
- Definiția inductivă:
  - $abb \in L$
  - Daca $X \in L$, atunci $aXbb \in L$
  - Niciun alt cuvânt nu face parte din L
- Definiția generativă:
- $G = (\{X\}, \{a, b\}, X, P)$, unde $P = \{X \longrightarrow aXbb, X \longrightarrow abb\}$
- Derivarea cuvântului $a^3 b^6$:
- $X \Rightarrow aXbb \Rightarrow a(aXbb)bb \Rightarrow aa(abb)bbbb$

Exemple (2)

- $L = \{a^n b^n c^n | n \geq 1\}$
- G= (N, T, S, P), $N = \{S, X\}$, $T = \{a, b, c\}$, P constă din:

  1. $S \rightarrow abc$
  2. $S \rightarrow aSXc$
  3. $cX \rightarrow Xc$
  4. $bX \rightarrow bb$

- Derivarea cuvântului $a^3 b^3 c^3$:

$$S \Rightarrow(2)\ aSXc \Rightarrow(2)\ aaSXcXc \Rightarrow(1)\ aaabcXcXc \Rightarrow(3)$$

$$aaabXccXc \Rightarrow(4)\ aaabbccXc \Rightarrow(3)\ aaabbcXcc \Rightarrow(3)$$

$$aaabbXccc \Rightarrow(4)\ aaabbbccc = a^3 b^3 c^3$$

Ierarhia lui Chomsky (1)

- **Avram Noam Chomsky**

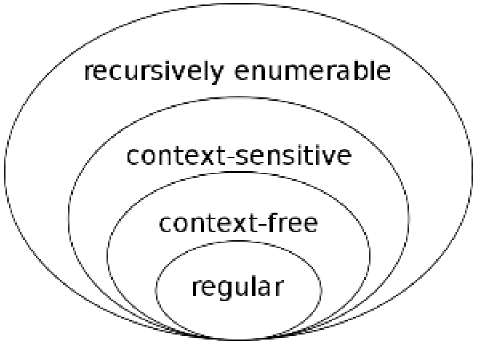
(n. 7 decembrie, 1928, Philadelphia, SUA)

- Este un lingvist și activist politic american, profesor emerit în lingvistică la Massachusetts Institute of Technology (MIT).
- În lumea academică, Chomsky este cunoscut pentru „teoria gramaticii generative” și pentru contribuțiile sale în domeniul lingvisticii teoretice.
- El este cel care a revoluționat întreg sistemul lingvistic modern prin celebrele sale modele generative.
- Supranumit „părintele lingvisticii moderne”

sursa: Wikipedia

Ierarhia lui Chomsky (2)

- **Gramatici de tip 0 (generale – recursiv enumerabile)**
  - $\alpha \rightarrow \beta$
  - Nu exista restrictii asupra regulilor



- **Gramatici de tip 1 (dependente de context)**
  - reguli de forma $\alpha\ A\ \beta \rightarrow \alpha\ \gamma\ \beta$ unde A $\in N$, $\gamma \neq \varepsilon$, α, $\beta \in (N \cup T)^*$, $S \rightarrow \varepsilon$, caz în care S nu apare în dreapta regulilor
- **Gramatici de tip 2 (independente de context)**
  - reguli de forma $A \rightarrow \gamma$ unde $A \in N$ și $\gamma \in (N \cup T)^*$
- **Gramatici de tip 3 (regulate)**
  - reguli $A \rightarrow a$ sau $A \rightarrow aB$ unde A, $B \in N$ și $a \in T^*$.

Clasificarea limbajelor

- Un limbaj L este de tipul j daca exista o gramatica G de tipul j astfel incat $L(G) = L$, unde $j \in \{0, 1, 2, 3\}$.
- Vom nota cu $\mathcal{L}_j$ clasa limbajelor de tipul j, unde $j \in \{0, 1, 2, 3\}$.
- Din ierarhia lui Chomsky: $L_3 \subset L_2 \subset L_1 \subset L_0$
- Incluziunile sunt stricte:
  - orice limbaj de tip $j + 1$ este si de tip $j \in \{0, 1, 2\}$
  - există limbaje de tip j care nu sunt de tip $j + 1$, $j \in \{0, 1, 2\}$

Proprietăți

- Fiecare din familiile $\mathcal{L}_j$ cu $0 \leq j \leq 3$ contine toate limbajele finite
- Fiecare din familiile $\mathcal{L}_j$ cu $0 \leq j \leq 3$ este inchisa la operatia de reuniune:

$$L_1, L_2 \in \mathcal{L}_j \Rightarrow L_1 \cup L_2 \in \mathcal{L}_j,$$

$$\forall j : 0 \leq j \leq 3$$

Gramatici de tip 3 (regulate)

- O gramatică $G = (N, T, S, P)$ este de tip 3 dacă regulile sale au forma:

  $A \to u$ sau $A \to uB$ unde $A, B \in N$ și $u \in T^*$.

- Exemplu:

  $G = (\{D\}, \{0, 1, ..., 9\}, D, P)$

  Unde P este:

  $D \to 0D|1D|2D| \ldots |9D$

  $D \to 0|1| \ldots |9$

Exemple

- Fie gramatica $G = (\{A, B\}, \{l, c\}, A, P)$ unde P este:
  - $A \longrightarrow lB$, $B \longrightarrow lB \,|\, cB \,|\, \varepsilon$ (l = litera, c = cifra)

Exemple

- Fie gramatica $G = (\{A, B\}, \{l, c\}, A, P)$ unde P este:
  - $A \rightarrow lB$, $B \rightarrow lB \,|\, cB \,|\, \varepsilon$ (l = litera, c = cifra)
- Ce ar putea reprezenta limbajul L(G)?
  - În legătură cu atomii recunoscuți de un compilator

Exemple

- Fie gramatica $G = (\{A, B\}, \{l, c\}, A, P)$ unde P este:
  - $A \rightarrow lB$, $B \rightarrow lB \mid cB \mid \varepsilon$ (l = litera, c = cifra)
- Ce ar putea reprezenta limbajul L(G)?
  - În legătură cu atomii recunoscuți de un compilator
- L(G) = Mulțimea identificatorilor
  - Denumiri de variabile, metode, constante, etc..
  - Încep obligatoriu cu o literă dar pot conține și cifre

Exemple

- Fie gramatica $G = (\{A, B\}, \{+, -, c\}, A, P)$ unde P este:
  - $A \longrightarrow +cB \mid -cB \mid cB$, $B \longrightarrow cB \mid \varepsilon$ (c = cifra)
- Ce reprezintă L(G)?

Exemple

- Fie gramatica $G = (\{A, B\}, \{+, -, c\}, A, P)$ unde P este:
  - $A \rightarrow +cB \mid -cB \mid cB$, $B \rightarrow cB \mid \varepsilon$ (c = cifra)
- Ce reprezintă L(G)?
  - $L(G)$: multimea constantelor intregi

Forma normală

- O gramatică de tip 3 este in formă normală dacă regulile sale sunt de forma $A \longrightarrow a$ sau $A \longrightarrow aB$, unde $a \in T$, si, eventual $S \longrightarrow \varepsilon$ (caz în care S nu apare în dreapta regulilor)
- Pentru orice gramatică de tip 3 există o gramatică echivalentă în formă normală.

Forma normală

- Obținerea gramaticii în formă normală echivalentă cu o gramatică de tip 3:
  - Se poate arăta că pot fi eliminate regulile de forma $A \rightarrow B$ (redenumiri) și cele de forma $A \rightarrow \varepsilon$ (reguli de ștergere), cu excepția eventual a regulii $S \rightarrow \varepsilon$.
  - Orice regulă de forma $A \rightarrow a_1 a_2 \ldots a_n$ se înlocuiește cu $A \rightarrow a_1 B_1$, $B_1 \rightarrow a_2 B_2$, . . ., $B_{n-2} \rightarrow a_{n-1} B_{n-1}$, $B_{n-1} \rightarrow a_n$, $n > 1$, $B_1, \ldots, B_{n-1}$ fiind neterminali noi.
  - Orice regulă de forma $A \rightarrow a_1 a_2 \ldots a_n B$ se înlocuiește cu $A \rightarrow a_1 B_1$, $B_1 \rightarrow a_2 B_2$, . . ., $B_{n-2} \rightarrow a_{n-1} B_{n-1}$, $B_{n-1} \rightarrow a_n B$, $n > 1$, $B_1, \ldots, B_{n-1}$ fiind neterminali noi
  - Transformările care se fac nu modifică limbajul generat de gramatică

Proprietăți de închidere pentru familia de limbaje regulate

- Fie L, L_1, L_2 limbaje de tip 3 (regulate). Atunci, urmatoarele limbaje sunt de asemenea de tip 3:
  - $L_1 \cup L_2$
  - $L_1 \cdot L_2$
  - L^*
  - L^R
  - $L_1 \cap L_2$
  - $L_1 \setminus L_2$

Închiderea la reuniune

- Fie L, L_1, L_2 limbaje de tip 3 (regulate).
- Fie $G_1 = (N_1, T_1, S_1, P_1)$ si $G_2 = (N_2, T_2, S_2, P_2)$ gramatici de tip 3 cu $L_1 = L(G_1)$, $L_2 = L(G_2)$.
- Presupunem $N_1 \cap N_2 = \emptyset$ si gramaticile în forma normala.

Atunci:

- Închiderea la reuniune: se arată că $L_1 \cup L_2 \in \mathcal{L}_3$:
- Gramatica $G = (N_1 \cup N_2 \cup \{S\}, T_1 \cup T_2, S, P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\})$ este de tip 3 si genereaza limbajul $L_1 \cup L_2$

Închiderea la operația de produs

- Fie L, L_1, L_2 limbaje de tip 3 (regulate).
- Fie $G_1 = (N_1, T_1, S_1, P_1)$ si $G_2 = (N_2, T_2, S_2, P_2)$ gramatici de tip 3 cu $L_1 = L(G_1)$, $L_2 = L(G_2)$.
- Presupunem $N_1 \cap N_2 = \emptyset$ si gramaticile în forma normala.

Atunci:

- Gramatica $G = (N_1 \cup N_2, T_1 \cup T_2, S_1, P)$ unde P constă din:
  - regulile de forma $A \to aB$ din P_1
  - reguli $A \to aS_2$ pentru orice regula de forma $A \to a$ din P_1
  - toate regulile din P_2
- este de tip 3 și generează limbajul L_1L_2.

Cursul viitor:

- Simplificarea gramaticilor independente de context
  - Eliminarea λ-producțiilor
  - Eliminarea producțiilor ciclice
  - Eliminarea recursivității stânga
  - Factorizarea stânga

Întrebări?