Κεφάλαιο 1: Οι Γραμματικές Chomsky

1.1 Ο Ρόλος της Γραμματικής

Μία γλώσσα προγραμματισμού ορίζεται σαν μία *τυπική γλώσσα* (*formal language*): είναι ακριβές πότε μία συμβολοσειρά (λέξη) ανήκει στην γλώσσα και πότε όχι.

- Μία μη τυπική γλώσσα: τα Ελληνικά!

Μας ενδιαφέρουν 2 είδη «εργαλείων»:

1. Αναγνώρισης Γλώσσας: ένας μηχανισμός που θα αποφασίζει (αυτόματα) αν μία λέξη ανήκει στην γλώσσα ή όχι (*αυτόματο* ή *compiler*).
2. Παραγωγός Γλώσσας: ένας μηχανισμός που σχηματίζει όλες τις λέξεις της γλώσσας (*γραμματική*).

- Αυτόματα και Γραμματικές είναι δύο όψεις του ίδιου νομίσματος.

Παράδειγμα: Συνήθως σε μία γλώσσα προγραμματισμού μία μεταβλητή αρχίζει με ένα γράμμα, ακολουθούμενο από ένα τυχαίο αριθμό γραμμάτων και ψηφίων (πχ AB12o, A, B2, …).

Μία γραμματική που παράγει όλες τις πιθανές μεταβλητές είναι η εξής:

Μετ ::= Γράμμα Σειρά
Σειρά ::= ε | Γράμμα Σειρά | Ψηφίο Σειρά
Γράμμα ::= A | B | … | Ω | α | β | … | ω
Ψηφίο ::= 0 | 1 | … | 9
⇒ Σημείωση: στα παραπάνω, ε είναι η κενή λέξη.

Ας δούμε τα συστατικά της γραμματικής ένα προς ένα:

- Τα σύμβολα που μπορούν να χρησιμοποιηθούν στην γλώσσα είναι: A, B, …, Ω, α, β, …, ω, 0, 1, …, 9. Αυτά λέγονται *τερματικά* ή *τελικά* σύμβολα.
- {Μετ, Σειρά, Γράμμα, Ψηφίο} είναι *μη τερματικά*. Χρησιμοποιούνται μόνο για την παραγωγή τελικών λέξεων. Μία λέξη είναι *τελική* όταν περιλαμβάνει μόνο τελικά σύμβολα. Είναι οι λέξεις που μπορούν να ανήκουν στην γλώσσα που μας ενδιαφέρει.
- Η παραγωγή αρχίζει με το σύμβολο Μετ, το *αρχικό σύμβολο*, που είναι πάντοτε μη τερματικό. Πχ το A123α παράγεται ως εξής:

Μετ	⇒	Γράμμα	Σειρά				
	⇒	A	Σειρά				
	⇒	A	Ψηφίο	Σειρά			
	⇒	A	Ψηφίο	Ψηφίο	Σειρά		
	⇒	A	Ψηφίο	Ψηφίο	Ψηφίο	Σειρά	
	⇒	A	1	Ψηφίο	Ψηφίο	Σειρά	
	⇒	A	1	2	Ψηφίο	Σειρά	
	⇒	A	1	2	3	Σειρά	
	⇒	A	1	2	3	Γράμμα	Σειρά
	⇒	A	1	2	3	α	Σειρά
	⇒	A	1	2	3	α	

- Ένας *κανόνας* έχει την μορφή u ::= v, όπου u, v είναι λέξεις και u≠ε.

Ένα σύνολο κανόνων:

$u ::= v_1$

$u ::= v_2$

…

$u ::= v_n$

μπορεί να συντομευτεί ως:

$u ::= v_1 \mid v_2 \mid \ldots \mid v_n$.

1.2 Ο Ορισμός της Γραμματικής

Μία γραμματική είναι μία τετράδα (N, T, P, S) όπου:

- N είναι ένα πεπερασμένο σύνολο μη τερματικών συμβόλων.
- T είναι ένα πεπερασμένο σύνολο τερματικών συμβόλων, όπου $N \cap T = \varnothing$.
- $S \in N$ είναι το αρχικό σύμβολο.
- P είναι ένα πεπερασμένο σύνολο κανόνων. Ένας κανόνας είναι ένα ζεύγος $(u,v) \in (N \cup T)^* \times (N \cup T)^*$ και γράφεται συνήθως ως $u ::= v$.
  Πρέπει:
  - ✓ $u \neq \varepsilon$
  - ✓ $u \notin T^*$ (δηλαδή αριστερά υπάρχει ένα τουλάχιστον μη τερματικό σύμβολο)

Ποια γλώσσα παράγει μία γραμματική G;

1. Έστω w, w′, x, y, u, $v \in (N \cup T)^*$. Το w′ *παράγεται άμεσα* από το w όταν w=xuy, w′=xvy και $(u ::= v) \in P$. Γράφουμε:
   $w \rightarrow_G w'$ (ή $w \rightarrow w'$).
2. Το w′ *παράγεται* από το w όταν $w = w_0 \rightarrow w_1 \rightarrow \ldots \rightarrow w_n = w'$. Γράφουμε:
   $w \rightarrow_G^* w'$ (ή $w \rightarrow^* w'$).
   Όταν μας ενδιαφέρει το μήκος της παραγωγής, γράφουμε:
   $w \rightarrow_G^n w'$ (ή $w \rightarrow^n w'$).
   Για λόγους «μαθηματικής ομορφιάς» γράφουμε:
   $w \rightarrow_G^0 w$ (ή $w \rightarrow^0 w$), για κάθε $w \in (N \cup T)^*$.
3. Η *γλώσσα που παράγεται από την γραμματική G* είναι το σύνολο όλων των τερματικών λέξεων που παράγονται από το αρχικό σύμβολο S:
   $L(G) = \{w \in T^* \mid S \rightarrow_G^* w\}$.

Παραδείγματα:

1. $S ::= aS \mid \varepsilon$
   $N = \{S\}$
   $T = \{a\}$
   $L(G) = \{a^n \mid n \geq 0\}$

2. $S ::= aSb \mid \varepsilon$
   $N = \{S\}$
   $T = \{a, b\}$
   $L(G) = \{a^n b^n \mid n \geq 0\}$

3. $S ::= aSb \mid a$
   $N = \{S\}$
   $T = \{a, b\}$
   $L(G) = \{a^{n+1} b^n \mid n \geq 0\}$

4. S ::= cS′
   S′ ::= aS′b | ε
   N={S, S′}
   T={a, b, c}
   L(G)={ca^nb^n | n≥0}

5. S ::= ASB | ε
   AB ::= BA
   A ::= a
   B ::= b
   N={S, A, B}
   T={a, b}
   L(G)={w∈{a,b}* | $count_a(w)=count_b(w)$}

6. S ::= aSBC | aBC
   CB ::= BC
   aB ::= ab
   bB ::= bb
   bC ::= bc
   cC ::= cc
   N={S, B, C}
   T={a, b, c}
   L(G)={$a^nb^nc^n$ | n≥1}

7. Program ::= Instruction_Sequence
   Instruction_Sequence ::= Instruction; Instruction_Sequence | ε
   Instruction ::= Assignment | Conditional | Loop
   Assignment ::= Var := Term
   Conditional ::= IF Condition THEN Program ELSE Program END
   Loop ::= WHILE Condition DO Program END
   Condition ::= (Var = Term) | (Var > Term) |
   (Condition AND Condition) |
   (Condition OR Condition) |
   (NOT Condition)
   Var ::= A | B | … | Z
   Term ::= …

Πώς αποδεικνύουμε L(G)=L;

1. **L⊆L(G)**: Δείχνουμε πως κάθε λέξη στο σύνολο L μπορεί να παραχθεί από την γραμματική G (συνήθως εύκολο).
2. **L(G)⊆L**: Η G δεν μπορεί να παράγει καμία τερματική λέξη που δεν ανήκει στο L (συνήθως δύσκολο). Συνήθης μέθοδος απόδειξης: επαγωγή ως προς το μήκος μιας παραγωγής.

Παραδείγματα:
Παράδειγμα 2
S ::= aSb | ε
L(G)={a^nb^n | n≥0}
"⊇": S→aSb→a^2Sb^2→…→a^nSb^n→a^nb^n.

"$\subseteq$": Θέλουμε να χρησιμοποιήσουμε μαθηματική επαγωγή στο μήκος μίας παραγωγής. Χρειαζόμαστε ένα πιο ισχυρό αποτέλεσμα:

<u>Λήμμα</u>

$S\rightarrow^{*}u\Rightarrow u=a^nSb^n$ ή $u=a^nb^n$.

<u>Απόδειξη</u>

(Επαγωγή)

$S\rightarrow^{0}u\Rightarrow u=S=a^0Sb^0$.

Έστω $S\rightarrow^{n}u\Rightarrow u=a^nSb^n$ ή $u=a^nb^n$.

$S\rightarrow^{n+1}v\Rightarrow$

$\Rightarrow S\rightarrow^{n}u\rightarrow v\Rightarrow$

$\Rightarrow u=a^nSb^n$ ή $u=a^nb^n$ (η περίπτωση $u=a^nb^n$ αποκλείεται αφού $u\rightarrow v)\Rightarrow$

$\Rightarrow S\rightarrow^{n}a^nSb^n\rightarrow v\Rightarrow$

$\Rightarrow v=a^nb^n$ ή $v=a^{n+1}Sb^{n+1}$.

Παράδειγμα 6

S ::= aSBC | aBC

CB ::= BC

aB ::= ab

bB ::= bb

bC ::= bc

cC ::= cc

$L(G)=\{a^nb^nc^n \mid n\geq 1\}$

"$\supseteq$": $S\rightarrow aSBC\rightarrow aaSBCBC\rightarrow\ldots\rightarrow$
$a^{n-1}S(BC)^{n-1}\rightarrow a^nBC(BC)^{n-1}\rightarrow$
$a^nBBC(BC)^{n-2}\rightarrow\ldots\rightarrow a^nB^nC^n=a^{n-1}aB^nC^n\rightarrow$
$a^{n-1}abB^{n-1}C^n\rightarrow a^nbbB^{n-2}C^n\rightarrow\ldots\rightarrow a^nb^nC^n\rightarrow$
$a^nb^ncC^{n-1}\rightarrow a^nb^nccC^{n-2}\rightarrow\ldots\rightarrow a^nb^nc^n$.

"$\subseteq$": Έστω $S\rightarrow^{*}u$. Τότε ισχύουν:

(1) $count_{\{a,A\}}(u)=count_{\{b,B\}}(u)=count_{\{c,C\}}(u)$

Γιατί; Επαγωγή!

Ισχύει για $S\rightarrow^{0}u\Rightarrow u=S$

Εάν ισχύει, η εφαρμογή οποιουδήποτε κανόνα διατηρεί την σχέση αυτή.

(2) $u=vaw\Rightarrow v=a^n$

(3) $u=vbw\Rightarrow count_{\{c,C,S\}}(v)=0$

Από (2), (3)$\Rightarrow u=a^nb^kc^j\ \forall u\in L(G)$ (4)

Από (1), (4)$\Rightarrow u=a^nb^nc^n\ \forall u\in L(G)$.

1.3 Η Ιεραρχία του Chomsky

Η ιεραρχία του Chomsky ορίζει διάφορους τύπους γραμματικών.

Γενικά ισχύει το εξής:

- Όσο πιο περιορισμένο το είδος των γραμματικών που επιτρέπει ένας τύπος, τόσο πιο εύκολος είναι ο προσδιορισμός κατά πόσον μία λέξη ανήκει στην γλώσσα που παράγεται από μία γραμματική του τύπου αυτού.

Τύποι γραμματικών

- *Τύπος 0*: Κανένας περιορισμός.

- *Τύπος 1*: (*Γραμματικές Ευαίσθητες στα Συμφραζόμενα – Context-sensitive Grammars*)
  Κάθε κανόνας έχει την μορφή:
  $u_1Au_2 ::= u_1wu_2$
  όπου $A\in N$, w, u_1, $u_2\in(N\cup T)^*$, $w\neq\varepsilon$.
- *Τύπος 2*: (*Γραμματικές Χωρίς Συμφραζόμενα – Context-free Grammars)*
  Κάθε κανόνας έχει την μορφή:
  $A ::= u$
  όπου $A\in N$, $u\in(N\cup T)^*$.
- *Τύπος 3*: (*Κανονικές Γραμματικές – Regular Grammars – Right Linear Grammars*)
  Κάθε κανόνας έχει μία από τις ακόλουθες μορφές:
  1. $A ::= u$, όπου $A\in N$, $u\in T^*$
  2. $A ::= uB$, όπου $A, B\in N$, $u\in T^+$, όπου $T^+=T^*-\{\varepsilon\}$.

Τύποι Γλωσσών

Μία γλώσσα L έχει τύπο i $\Leftrightarrow$ υπάρχει γραμματική G τύπου i ώστε L=L(G).

<u>Θεώρημα 1.1</u> Έστω $\mathcal{L}_i=\{L \mid L$: γλώσσα του τύπου i$\}$. Τότε:

1. $\mathcal{L}_1\subset\mathcal{L}_0$.
2. $\{L-\{\varepsilon\} \mid L\in\mathcal{L}_2\}\subset\mathcal{L}_1$.
3. $\mathcal{L}_3\subset\mathcal{L}_2$.

<u>Δεν</u> θα δώσουμε απόδειξη για αυτό το θεώρημα. Παρόλα αυτά θα δείξουμε τουλάχιστον ορισμένα μέρη του.

<u>Παρατηρήσεις</u>

1. $\mathcal{L}_1\subseteq\mathcal{L}_0$, $\mathcal{L}_3\subseteq\mathcal{L}_2$: προφανή.
2. $\varepsilon\in L\Rightarrow L\notin\mathcal{L}_1$ (διότι ξεκινάμε με το S και η δεξιά πλευρά κάθε κανόνα είναι εξ ορισμού μακρύτερη από την αριστερή). Γι' αυτό το 2 του θεωρήματος έχει αυτή την μορφή.

1.4 Γλώσσες Τύπου 0

Χωρίς κανένα περιορισμό ισχύει το εξής:

- Δεν μπορούμε να αποφασίσουμε αν $w\in L(G)$.

Είναι ένα από τα προβλήματα που δεν μπορούν να επιλυθούν με προγραμματισμό. Θα εξετάσουμε τέτοια προβλήματα αργότερα στο μάθημα.
Αλλά τουλάχιστον μπορούμε να λύσουμε το «μισό πρόβλημα». Θα δείξουμε ότι για κάθε γραμματική G μπορούμε να απαριθμήσουμε όλες τις λέξεις στο L(G).

$\Rightarrow$ Έστω $w\in T^*$. Θέλουμε να ξέρουμε αν $w\in L(G)$. Απαριθμούμε όλο το L(G):

- Αν $w\in L(G)$, θα εμφανιστεί σε πεπερασμένο χρόνο.
- Αν $w\notin L(G)$, τότε θα περιμένουμε έπ' άπειρον (δεν μπορούμε να ξέρουμε αν το w θα απαριθμηθεί στο μέλλον).

Η Απαρίθμηση του L(G)

Έστω G=(N, T, P, S) μία τυχαία γραμματική.

- $S(G)_k := \{w \mid S \rightarrow^n w, n \leq k\}$.
  Το $S(G)_k$ περιέχει όλες τις λέξεις που η G μπορεί να παράγει σε το πολύ k βήματα.
- $S(G)_0 = \{S\}$.
- $S(G)_{k+1} = S(G)_k \cup \{w \mid v \rightarrow w, v \in S(G)_k\}$.
- $S(G)_k$ είναι πεπερασμένα: επαγωγή ως προς k.
- $S(G)_{k+1}$ μπορεί να υπολογιστεί από το $S(G)_k$ μέσω ενός προγράμματος.

$\Rightarrow S(G) = \cup_{k \in \mathbb{N}} S(G)_k$ είναι απαριθμήσιμο.
Αλλά: $L(G) = S(G) \cap T^*$.
$\Rightarrow$ L(G): απαριθμήσιμο. Φιλτράρουμε όλα τα $u \in S(G)$ που δεν είναι τερματικά.

Με πιο απλά λόγια:

- ✓ Πρώτα απαριθμούμε το S.
- ✓ Μετά όλες τις λέξεις που παράγει η G σε ένα βήμα.
- ✓ Μετά όλες τις λέξεις που παράγει η G σε δύο βήματα.
- ✓ …
- ✓ Μετά όλες τις λέξεις που παράγει η G σε n βήματα.
- ✓ …

1.5 Γλώσσες Τύπου 1

Μονότονες Γραμματικές

Κάθε κανόνας έχει την μορφή:
u ::= v
όπου $|u| \leq |v|$.

Προφανώς ισχύει: $w \rightarrow^* w' \Rightarrow |w| \leq |w'|$.
(Απόδειξη με μαθηματική επαγωγή στο μήκος της παραγωγής).

Προφανώς κάθε γραμματική τύπου 1 είναι μονότονη. Αλλά ισχύει και το εξής: για κάθε μονότονη γραμματική G υπάρχει μία γραμματική G′ τύπου 1 με L(G)=L(G′) (χωρίς απόδειξη). Άρα έχουμε:
Θεώρημα 1.2 Οι μονότονες γραμματικές παράγουν ακριβώς τις γλώσσες τύπου 1.

- Τι κερδίσαμε σε σχέση με τον τύπο 0; Πολλά!

Έστω $w \in T^*$ με $|w|=k$. Θέλουμε να αποφασίσουμε εάν $w \in L(G)$. Εάν κατά την απαρίθμηση της L(G) μία παραγωγή δώσει u με $|u|>k$, μπορούμε να σταματήσουμε την συγκεκριμένη παραγωγή. Αλλά οι λέξεις $u \in L(G)$ με $|u| \leq k$ είναι πεπερασμένες, και μπορούμε να υπολογίσουμε πότε όλες πρέπει να έχουν εξαντληθεί. Εάν ως τότε το w δεν απαριθμήθηκε, τότε αποφασίζουμε: $w \notin L(G)$!
Τώρα θα γράψουμε αυτό το επιχείρημα με μαθηματικό τρόπο:
Θεώρημα 1.3 Έστω G=(N, T, P, S) μία μονότονη γραμματική. Τότε είναι επιλύσιμο το κατά πόσον $w \in L(G)$.
Απόδειξη
Έστω $|w|=k \geq 1$ ($\varepsilon \notin L(G)$ για κάθε μονότονη γραμματική G).
Θέτω:
$T_n^m = \{u \in (N \cup T)^* \mid |u| \leq n, S \rightarrow^j u, j \leq m\}$.

Το T_n^m αποτελείται από όλες τις λέξεις που παράγονται το πολύ σε m βήματα και έχουν μήκος το πολύ n.
Ισχύει:

- $T_n^0 = \{S\}$.
- $T_n^{m+1} = T_n^m \cup \{u \in (N \cup T)^* \mid w \rightarrow u, w \in T_n^m, |u| \leq n\}$.

Έχουμε:

1. $\{w \in L(G) \mid |w| \leq n\} \subseteq \cup_{m \geq 0} T_n^m$.
2. $T_n^0 \subseteq T_n^1 \subseteq T_n^2 \subseteq \ldots$
3. Υπάρχει πεπερασμένος αριθμός λέξεων μήκους το πολύ n στο $N \cup T$, άρα:
   $\exists m: T_n^m = T_n^{m+1} = T_n^{m+2} = \ldots$

Άρα υπολογίζουμε τα σύνολα T_k^m εώς ότου είτε $w \in T_k^m$ είτε $T_k^m = T_k^{m+1}$.

Αλγόριθμος

```
k := |w|;
m := 0;
T_k^0 := {S};
REPEAT
    Υπολόγισε T_k^{m+1};
    m := m+1;
UNTIL w∈T_k^m OR T_k^{m−1}=T_k^m;
IF w∈T_k^m
    THEN "w∈L(G)"
    ELSE "w∉L(G)"
END.
```