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

Σ simboli terminali

M variabili o meta simboli

S assioma

P regole della forma $\alpha \to \beta$

$\Rightarrow$ applicazione di 1 regola

$\Rightarrow^*$ " di più regole

$$L(G) = \{ \underbrace{w \in \Sigma^*}_{\text{parole di simboli terminali}} \mid S \Rightarrow^* w \}$$

Domanda: Tutti i linguaggi ammettono una grammatica?

Risposta: No!

Teorema: L è ricorsivamente enumerabile

$\Updownarrow$

L è generato da una grammatica

dimostrazione $\Uparrow$:

Data una grammatica G per L, posso costruire una procedura w t.c.

$$F_w(x) = \begin{cases} 1 & \text{se } x \in L \\ \uparrow & \text{se } x \notin L \end{cases}$$

quindi w termina e dà 1 per le parole generate da G, sulle altre non termina

Definizioni

$F_i = \{$ parole di simboli terminali e meta simboli ottenute da S in "i" passi di derivazione $\}$

$T_i : \{$ parole di soli simboli terminali ottenute da S in "i" passi di derivazione $\}$

es $S \Rightarrow w_1 \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \dots \Rightarrow w_i$

$$\underset{F_1}{w_1} \quad \underset{F_2}{w_2} \quad \underset{F_3}{w_3} \quad \underset{F_i}{w_i}$$

Se $w_i \in \Sigma^* \Rightarrow w_i \in T_i$

Nota: $T_i \subseteq F_i$

Fatto: $L(G) = \bigcup_i T_i$

Infatti:

(1) $L(G) \subseteq \bigcup_i T_i$

$x \in L(G) \Rightarrow$ ho $S \Rightarrow^* x$ in $G \Rightarrow$

$\Rightarrow \exists\, i$ numero di passi di derivazione per $S \Rightarrow^* x \Rightarrow x \in T_i \Rightarrow$

$\Rightarrow x \in \bigcup_i T_i$

(2) $\bigcup_i T_i \subseteq L(G)$

$x \in \bigcup_i T_i \Rightarrow \exists\, i$ t.c. $x \in T_i$

$\Rightarrow$ ho $S \Rightarrow^* x$ in G che consiste di i passi

$\Rightarrow$ ho $S \Rightarrow^* x$ in $G \Rightarrow x \in L(G)$

Formalizzo:

$$F_i = \{ \gamma \in (\Sigma \cup M)^* \mid \eta \Rightarrow \gamma \text{ e } \eta \in F_{i-1} \}$$

$$T_i = \{ x \in \Sigma^* \mid x \in F_i \}$$

Inizialmente costruisco una procedura ELENCA che genera in maniera sistematica le parole di $L(G) = \bigcup_i T_i$

ELENCA stampa: T_1
T_2
T_3
$\vdots$

se $L(G)$ è infinito la procedura continua all'infinito

Introdurre input x

```
Procedure ELENCA() {

    G = (Σ, M, S, P);   // Fisso G
    F0 = { S };
    i = 1;
    while (i > 0) do {
        Fi = costruisciF(Fi-1, G);
        Ti = costruisciT(Fi, G);
        OUTPUT(Ti)
        i++;
    }
}
```

if cerco la parola in T_i

*

Vediamo le funzioni di ELENCA:

```
COSTRUISCIF ( F_{i-1} , G ) {

    F_i = ∅

    foreach η ∈ F_{i-1} do
        foreach α -> β ∈ P do
            Foreach x, y ∈ (Σ ∪ M)* t.c. η = xαy do
                F_i = F_i ∪ { xβy }

    return (F_i);
}
```

```
COSTRUISCIT ( F_i , G ) {
    T_i = ∅

    Foreach w ∈ F_i do
        if w ∈ Σ* then
            T_i = T_i ∪ {w}

    return(T_i)
}
```

* Adesso trasformo ELENCA nella procedura di ricerca per dim. che $L(G)$ è ricorsivamente enumerabile:

I modifica:

Passo $x \in \Sigma^*$ in input ad ELENCA

II modifica:

Sostituisco output(T_i) con
if$(x \in T_i)$ then return(1)

Correttezza di ELENCA modificato:

$x \in L(G) \Rightarrow \exists\, i$ t.c. $x \in T_i$

$\Rightarrow$ il ciclo while permette l'esecuzione delle if $(x \in T_i)$ e viene eseguita return(1)

$\Rightarrow$ ELENCA$(x) = 1$

$x \notin L(G) \Rightarrow x \notin \bigcup_i T_i \Rightarrow$

$\Rightarrow \forall i \;\; x \in T_i$

$\Rightarrow$ il ciclo while andrà in loop

$\Rightarrow$ ELENCA$(x) \uparrow$