Hoy: Context-Free grammars (CFG)

Intuitivamente, una (CFG) es un conjunto de reglas que permiten manipular strings

Ejemplo: $S \longrightarrow aMb$ ("se puede reemplazar con")

$M \longrightarrow A$

$M \longrightarrow B$

$A \longrightarrow aA$

$B \longrightarrow Bb$

$A \longrightarrow \varepsilon$

$B \longrightarrow \varepsilon$

empezando con S

Un posible uso de las reglas sería así:

$$S \to aMb \to aAb \to aaAb \to aaaAb \to aaa\varepsilon b$$

No tiene ningún símbolo de la izquierda

$aaab$

no se puede manipular más

Otra posibilidad

$$S \to aMb \to aBb \to aBbb \to a\varepsilon bb = abb$$

abb

¿Cuál es el lenguaje generado por este CFG?

↑ las palabras terminales, (las que ya no podemos modificar más)

Respuesta: es $a(a^* \cup b^*)b \subseteq \{a,b\}^*$

Viendo nuestro ejemplo queremos permitir una colección de símbolos "intermedios" que estan a la izquierda de las reglas y una de símbolos finales, que NO puedan aparecer a la izquierda así como una colección de reglas de sustitución. Más precisamente

Def: Un context-free gramar CFG es una 4-tupla (V, Σ, R, S) con:

V — conjunto finito

$V \supseteq \Sigma$ — símbolos terminales

$R \subseteq (V \setminus \Sigma) \times V^*$ — colección de reglas de sustitución

$V \ni S$ — estado de inicio

$(A, w) \in R$ se piensa como $A \to w \in V^*$

es válido reemplazar una instancia de A por w.

Cómputo:

Para $u, v \in V^*$ definimos

$u \Rightarrow v$ si es posible llegar desde u a v mediante una sustitución válida, es decir:

$$\left[\exists\, x, y \in V^* : \; u = x u' y \;\; v = x v' y \;\; y \;\; u' \to v' \in R \;\; ((u', v') \in R) \right]$$
$$u' \in V \setminus \Sigma, \quad v' \in V^*$$

$u \Rightarrow^* v$ si es posible llegar de u a v mediante un número finito de sustituciones válidas ($\Rightarrow^*$ es la clausura reflexiva y transitiva de $\Rightarrow$)

Def: El lenguaje ~~generado~~ aceptado por G consiste de las palabras en símbolos terminales que podamos escribir empezando desde el símbolo de inicio S.

$$L(G) = \{ w \in \Sigma^* : S \Rightarrow^* w \} \subseteq \Sigma^*$$

En aplicaciones de CFG hay dos problemas de mucha importancia:

(i) Capacidad de reconocer los strings de $L(G)$

(ii) Capacidad de generar los strings de $L(G)$

(Expresiones balanceadas)

R:

Ejemplo:

$S \to \varepsilon$

$S \to SS$

$S \to (S)$

$V = \{S, (,)\}$

$\Sigma = \{(,)\}$

$S \to SS \to S(S) \to S(SS) \to S(S(S)) \to (S)(S(S)) \to ()(())$

expresiones válidas en paréntesis

Ejemplo: $V = \{+, *, (,), v, T, F, E\}$

Σ

T: term

F: factor

E: expresión

Símbolo inicial

R:

$E \to E+T$

$E \to T$

$T \to T*F$

$T \to F$

$F \to (E)$

$F \to v$

Expresiones algebraicas $(v*v + v)*(v+v)$ ✓

Ejemplo: $V = \{S, 0, 1\}$, $\Sigma = \{0, 1\}$,

R: $S \to 0S1$

$S \to \varepsilon$

$L(G) = \{0^n 1^n : n \in \mathbb{N}\}$

así que hay lenguajes NO Regulares que pueden ser generados por un context-free gramm.

Teorema: $\{$Lenguajes regulares$\} \subseteq \{$Lenguajes generados por un CFG$\}$