Parsing: cómputo de First y Follow

Clase 22

IIC 2223

Prof. Cristian Riveros

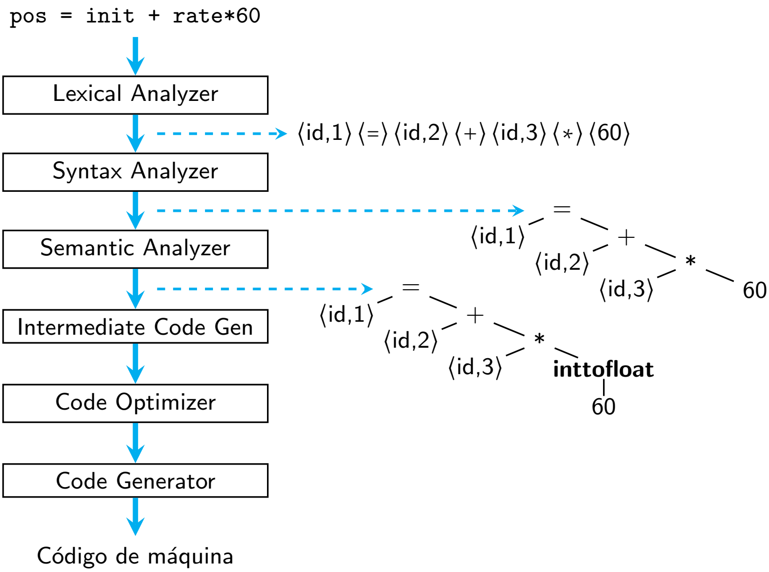
Sintaxis y semántica de un lenguaje (recordatorio)

Definición

1. La **sintaxis** de una lenguaje es un conjunto de reglas que describen los programas válidos que tienen significado.

2. La **semántica** de un lenguaje define el significado de un programa correcto según la sintaxis.

La estructura de un compilador (recordatorio)



Verificación de sintaxis

En este proceso se busca:

- verificar la sintaxis de un programa.
- entregar la estructura de un programa (árbol de parsing).

Consta de tres etapas:

1. Análisis léxico (**Lexer**). ✓
2. Análisis sintáctico (**Parser**).
3. Análisis semántico.

Ahora veremos como hacer el **Parser**.

Análisis sintáctico (Parser)

Informalmente

> *"Dado una secuencia de tokes w' y una* gramática $\mathcal{G}$ *construir un árbol de derivación (parsing) de $\mathcal{G}$ para w."*

Con el **árbol de derivación** habremos:

- verificado la sintaxis.
- obtenido la estructura.

Análisis sintáctico (Parser)

Ejemplo de gramática

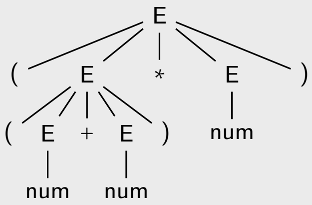
$$E \quad \rightarrow \quad (\, E + E \,) \mid (\, E * E \,) \mid \mathsf{num}$$

Para un input $w = ((43 + 56) * 27)$:

- Convertimos w en una secuencia de **tokens**:

$$w' \;=\; ((\mathsf{num} + \mathsf{num}) * \mathsf{num})$$

- Construimos un árbol de **parsing** para w':



Análisis sintáctico (Parser)

Problema de parsing

Dado una palabra w y dado una gramática $\mathcal{G}$,
generar un árbol de parsing $\mathcal{T}$ de $\mathcal{G}$ para w.

¿ya sabemos resolver este problema?

El algoritmo CKY nos permite hacer esto, pero ...

- impracticable para grandes inputs.
- multiples pasadas sobre el input.

Deseamos hacer parsing en **tiempo lineal** en el tamaño del input.

Autómatas apiladores al rescate (recordatorio)

Para una gramática $\mathcal{G} = (V, \Sigma, P, S)$
podemos construir un PDA alternativo $\mathcal{D}$ que acepta $\mathcal{L}(\mathcal{G})$:

$$\mathcal{D} = \big(V \cup \Sigma \cup \{q_0, q_f\}, \Sigma, \Delta, q_0, \{q_f\}\big)$$

La relación de transición Δ se define como:

$$\begin{array}{rclcl}
\Delta & = & \{\, (q_0, \epsilon, S \cdot q_f) \,\} & \cup & \\
 & & \{\, (X, \epsilon, \gamma) \mid X \to \gamma \in P \,\} & \cup & (\textbf{Expandir}) \\
 & & \{\, (a, a, \epsilon) \mid a \in \Sigma \,\} & & (\textbf{Reducir})
\end{array}$$

¿cómo usamos este PDA para hacer parsing?

Problema: muchas alternativas para **expandir**.

¿cómo elegir la siguiente producción para expandir?

$$X \quad \rightarrow \quad \alpha \mid \beta$$

¿cómo elegir entre α o β?

Queremos elegir la **próxima producción** $X \rightarrow \gamma$ de tal manera que, si existe una derivación para el input, entonces $X \rightarrow \gamma$ **es parte de esa derivación**:

$$\text{si} \quad S \overset{\star}{\underset{\text{lm}}{\Rightarrow}} uX\gamma' \overset{\star}{\underset{\text{lm}}{\Rightarrow}} uv, \quad \text{entonces} \quad \gamma\gamma' \overset{\star}{\underset{\text{lm}}{\Rightarrow}} v$$

Necesitamos **mirar las siguientes letras en** v y
ver si pueden ser producidas por α o β.

...para esto ocuparemos `first` y `follow`.

Outline

Outline

Prefijos

First y Follow

Calcular First

Calcular Follow

Definiciones de prefijos

Definiciones

Sea Σ un alfabeto finito. Para un $k \geq 0$, se define:

$$\begin{aligned} \Sigma^{\leq k} &= \bigcup_{i=0}^{k} \Sigma^{i} \\ \Sigma_{\#}^{\leq k} &= \Sigma^{\leq k} \cup \left(\Sigma^{\leq k-1} \cdot \{\#\}\right) \end{aligned}$$

Ejemplos

Para $\Sigma = \{a, b\}$:

- $\Sigma^{\leq 2} = \{\epsilon, a, b, aa, ab, ba, bb\}$
- $\Sigma_{\#}^{\leq 2} = \{\epsilon, a, b, aa, ab, ba, bb\} \cup \{\#, a\#, b\#\}$

Símbolo $\#$ representará un `EOF` (End of File),
marcando el **fin de una palabra**.

Definiciones de prefijos

Definiciones

Sea Σ un alfabeto finito. Para un $k \geq 0$, se define:

$$
\begin{aligned}
\Sigma^{\leq k} &= \bigcup_{i=0}^{k} \Sigma^i \\
\Sigma_{\#}^{\leq k} &= \Sigma^{\leq k} \cup \left(\Sigma^{\leq k-1} \cdot \{\#\}\right)
\end{aligned}
$$

Para una palabra $w = a_1 a_2 \ldots a_n \in \Sigma^*$ se define el **k-prefijo** de w como:

$$
w|_k = \begin{cases} a_1 \ldots a_n & \text{si } n \leq k \\ a_1 \ldots a_k & \text{si } k < n \end{cases}
$$

Definimos la **k-concatenación** $\odot_k$ entre strings $u, v \in \Sigma$ como:

$$
u \odot_k v = (u \cdot v)|_k
$$

Definiciones de prefijos

Definiciones

Para una palabra $w = a_1 a_2 \dots a_n \in \Sigma^*$ se define el **k-prefijo** de w como:

$$w|_k \;=\; \begin{cases} a_1 \dots a_n & \text{si } n \leq k \\ a_1 \dots a_k & \text{si } k < n \end{cases}$$

Definimos la **k-concatenación** $\odot_k$ entre strings $u, v \in \Sigma$ como:

$$u \odot_k v \;=\; (u \cdot v)|_k$$

Ejemplos

- $(abaa)|_2 = ab \qquad (ab)|_2 = ab \qquad (a)|_2 = a \qquad (\epsilon)|_2 = \epsilon$
- $a \odot_2 baa = (abaa)|_2 = ab$
- $bba \odot_2 a = (bbaa)|_2 = bb$
- $b \odot_2 \epsilon = (b)|_2 = b$

Definiciones de prefijos

Definiciones

Para una palabra $w = a_1 a_2 \dots a_n \in \Sigma^*$ se define el **k-prefijo** de w como:

$$w|_k \;=\; \begin{cases} a_1 \dots a_n & \text{si } n \leq k \\ a_1 \dots a_k & \text{si } k < n \end{cases}$$

Definimos la **k-concatenación** $\odot_k$ entre strings $u, v \in \Sigma$ como:

$$u \odot_k v \;=\; (u \cdot v)|_k$$

Extendemos estas operaciones para lenguajes $L, L_1, L_2 \subseteq \Sigma^*$ como:

$$\begin{aligned} L|_k &= \{w|_k \mid w \in L\} \\ L_1 \odot_k L_2 &= \{w_1 \odot_k w_2 \mid w_1 \in L_1 \text{ y } w_2 \in L_2\} \end{aligned}$$

Definiciones de prefijos

Definiciones

$$w|_k = \begin{cases} a_1 \dots a_n & \text{si } n \leq k \\ a_1 \dots a_k & \text{si } k < n \end{cases} \qquad L|_k = \{w|_k \mid w \in L\}$$

$$u \odot_k v = (u \cdot v)|_k \qquad L_1 \odot_k L_2 = \{w_1 \odot_k w_2 \mid w_1 \in L_1 \text{ y } w_2 \in L_2\}$$

Ejemplos

- $((ab)^*)|_3 = \{\epsilon, ab, aba\}$
- $(a)^* \odot_3 (ab)^* = \{\epsilon, a, aa, aaa, ab, aba, aab\}$

Los operadores $|_k$ y $\odot_k$ "miran" hasta un prefijo k.

Algunas propiedades de k-prefijos

Definiciones

$$w|_k \;=\; \begin{cases} a_1 \ldots a_n & \text{si } n \leq k \\ a_1 \ldots a_k & \text{si } k < n \end{cases} \qquad\qquad L|_k \;=\; \{w|_k \mid w \in L\}$$

$$u \odot_k v \;=\; (u \cdot v)|_k \qquad\qquad L_1 \odot_k L_2 \;=\; \{w_1 \odot_k w_2 \mid w_1 \in L_1 \text{ y } w_2 \in L_2\}$$

Propiedades

Para todo $k \geq 1$ y $L_1, L_2, L_3 \subseteq \Sigma^*$:

1. $L_1 \odot_k (L_2 \odot_k L_3) \;=\; (L_1 \odot_k L_2) \odot_k L_3$
2. $L_1 \odot_k \{\epsilon\} \;=\; \{\epsilon\} \odot_k L_1 \;=\; L_1|_k$
3. $(L_1 L_2)|_k \;=\; L_1|_k \odot_k L_2|_k$
4. $L_1 \odot_k (L_2 \cup L_3) \;=\; (L_1 \odot_k L_2) \cup (L_1 \odot_k L_3)$

Demostración: ejercicio.

Outline

Definición de $\texttt{first}_k$ y $\texttt{follow}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Definiciones

Se define la función $\texttt{first}_k : (V \cup \Sigma)^* \to 2^{\Sigma^{\leq k}}$ tal que, para $\gamma \in (V \cup \Sigma)^*$:

$$\texttt{first}_k(\gamma) \;=\; \{u|_k \mid \gamma \overset{\star}{\Rightarrow} u\}$$

Ejemplos

$$E \;\to\; (\,E + E\,) \mid (\,E * E\,) \mid \mathsf{n}$$

- $\texttt{first}_1(E) \;=\; \{\,(\,,\,\mathsf{n}\,\}$
- $\texttt{first}_2(E) \;=\; \{\,\mathsf{n}\,,\,(\mathsf{n}\,,\,((\,\}$
- $\texttt{first}_3(E) \;=\; \{\,\mathsf{n}\,,\,(\mathsf{n}+\,,\,(\mathsf{n}*\,,\,((\mathsf{n}\,,\,(((\,\}$

Definición de $\texttt{first}_k$ y $\texttt{follow}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Definiciones

Se define la función $\texttt{first}_k : (V \cup \Sigma)^* \to 2^{\Sigma^{\leq k}}$ tal que, para $\gamma \in (V \cup \Sigma)^*$:

$$\texttt{first}_k(\gamma) \;=\; \{u|_k \mid \gamma \overset{\star}{\Rightarrow} u\}$$

Se define la función $\texttt{follow}_k : V \to 2^{\Sigma_{\#}^{\leq k}}$ como:

$$\texttt{follow}_k(X) \;=\; \{w \mid S \overset{\star}{\Rightarrow} \alpha X \beta \;\text{ y }\; w \in \texttt{first}_k(\beta\#)\}$$

Definición de $\texttt{first}_k$ y $\texttt{follow}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Definiciones

$$\begin{aligned} \texttt{first}_k(\gamma) &= \{u|_k \mid \gamma \overset{\star}{\Rightarrow} u\} \\ \texttt{follow}_k(X) &= \{w \mid S \overset{\star}{\Rightarrow} \alpha X \beta \text{ y } w \in \texttt{first}_k(\beta\#)\} \end{aligned}$$

Ejemplos

$$E \quad \rightarrow \quad (\, E + E \,) \mid (\, E * E \,) \mid \mathsf{n}$$

- $\texttt{follow}_1(E) = \{\#, +, *,)\}$
- $\texttt{follow}_2(E) = \{\, \#\, ,\ )\#\, ,\ ))\, ,\ )+\, ,\ )*, +(, *(, +n, *n \,\}$

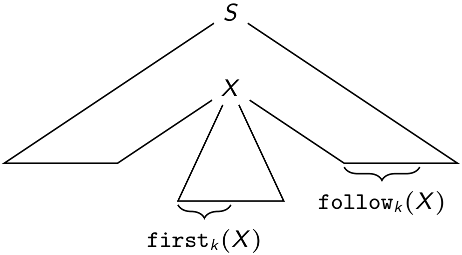
Definición de $\mathtt{first}_k$ y $\mathtt{follow}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Definiciones

$$\mathtt{first}_k(\gamma) = \{u|_k \mid \gamma \overset{\star}{\Rightarrow} u\}$$

$$\mathtt{follow}_k(X) = \{w \mid S \overset{\star}{\Rightarrow} \alpha X \beta \text{ y } w \in \mathtt{first}_k(\beta\#)\}$$



Outline

Propiedades de $\texttt{first}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Proposición

Para $X_1, \ldots, X_n \in V \cup \Sigma$:

$$\texttt{first}_k(X_1 \ldots X_n) \;=\; \texttt{first}_k(X_1) \odot_k \cdots \odot_k \texttt{first}_k(X_n)$$

Demostración

Defina $\mathcal{L}(X) = \{w \mid X \overset{\star}{\Rightarrow} w\}$ y $\mathcal{L}(\gamma) = \{w \mid \gamma \overset{\star}{\Rightarrow} w\}$.

Notar que $\texttt{first}_k(\gamma) = \mathcal{L}(\gamma)|_k$.

Por lo tanto, tenemos que:

$$\begin{array}{rcl} \texttt{first}_k(X_1 \ldots X_n) & = & \mathcal{L}(X_1 \ldots X_n)|_k \\ & = & (\mathcal{L}(X_1) \cdot \mathcal{L}(X_2) \cdot \ldots \cdot \mathcal{L}(X_n))|_k \\ & = & \mathcal{L}(X_1)|_k \odot_k \mathcal{L}(X_2)|_k \odot_k \ldots \odot_k \mathcal{L}(X_n))|_k \\ & = & \texttt{first}_k(X_1) \odot_k \cdots \odot_k \texttt{first}_k(X_n) \end{array}$$

■

Propiedades de $\texttt{first}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Proposición

Para $X_1, \ldots, X_n \in V \cup \Sigma$:

$$\texttt{first}_k(X_1 \ldots X_n) \;=\; \texttt{first}_k(X_1) \odot_k \cdots \odot_k \texttt{first}_k(X_n)$$

En particular, tenemos que:

$$\texttt{first}_k(X) \;=\; \bigcup_{X \to X_1 \ldots X_n \in P} \texttt{first}_k(X_1) \odot_k \cdots \odot_k \texttt{first}_k(X_n)$$

Propiedades de $\texttt{first}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

En particular, tenemos que:

$$\texttt{first}_k(X) \;=\; \bigcup_{X \to X_1 \ldots X_n \in P} \texttt{first}_k(X_1) \odot_k \cdots \odot_k \texttt{first}_k(X_n)$$

Defina el siguiente **programa recursivo** para todo $X \in V \cup \Sigma$:

$$\begin{aligned}
\texttt{first}_k^0(X) &:= \bigcup_{X \to w \in P} w|_k \\
\texttt{first}_k^i(X) &:= \bigcup_{X \to X_1 \ldots X_n \in P} \texttt{first}_k^{i-1}(X_1) \odot_k \cdots \odot_k \texttt{first}_k^{i-1}(X_n)
\end{aligned}$$

Es fácil ver que:

- $\texttt{first}_k^{i-1}(X) \subseteq \texttt{first}_k^i(X)$ para todo $i > 1$.
- Como $\texttt{first}_k(X) \subseteq \Sigma^{\leq k}$, entonces para algún $i \leq k \cdot |\Sigma|^k \cdot |V|$ tendremos:

$$\texttt{first}_k^j(X) = \texttt{first}_k^{j+1}(X) \text{ para todo } j \geq i$$

¿cómo calcular $\texttt{first}_k$?

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Defina el siguiente **programa recursivo** para todo $X \in V \cup \Sigma$:

$$
\begin{array}{lcl}
\texttt{first}_k^0(X) & := & \displaystyle\bigcup_{X \to w \in P} w|_k \\
\texttt{first}_k^i(X) & := & \displaystyle\bigcup_{X \to X_1 \dots X_n \in P} \texttt{first}_k^{i-1}(X_1) \odot_k \cdots \odot_k \texttt{first}_k^{i-1}(X_n)
\end{array}
$$

Teorema

Sea i^* el menor número tal que $\texttt{first}_k^{i^*}(X) = \texttt{first}_k^{i^*+1}(X)$ para todo $X \in V$. Entonces para todo $X \in V$:

$$\texttt{first}_k^{i^*}(X) \;=\; \texttt{first}_k(X)$$

Demostración (idea)

- $\subseteq$ Por inducción, demostrar que $\texttt{first}_k^i(X) \subseteq \texttt{first}_k(X)$.
- $\supseteq$ Por inducción, si $X \overset{\star}{\Rightarrow} w$, entonces $w|_k \in \texttt{first}_k^i(X)$ para algún i.

¿cómo calcular $\mathtt{first}_k$?

input : Gramática $\mathcal{G} = (V, \Sigma, P, S)$ y $k \geq 1$

output: Todos los conjuntos $\mathtt{first}_k(X)$ para todo $X \in V \cup \Sigma$.

Function `CalcularFIRST` *($\mathcal{G}$, k)*

- **foreach** $a \in \Sigma$ **do**
  - $\mathtt{first}_k^0(a) := \{a\}$
- **foreach** $X \in V$ **do**
  - $\mathtt{first}_k^0(X) := \bigcup_{X \to w \in P} w|_k$
- $i := 0$
- **repeat**
  - $i := i + 1$
  - **foreach** $a \in \Sigma$ **do**
    - $\mathtt{first}_k^i(a) := \{a\}$
  - **foreach** $X \in V$ **do**
    - $\mathtt{first}_k^i(X) := \bigcup_{X \to X_1 \dots X_n \in P} \mathtt{first}_k^{i-1}(X_1) \odot_k \cdots \odot_k \mathtt{first}_k^{i-1}(X_n)$
- **until** $\mathtt{first}_k^i(X) = \mathtt{first}_k^{i-1}(X)$ para todo $X \in V \cup \Sigma$
- **return** $\{\mathtt{first}_k(X)\}_{X \in V \cup \Sigma}$

Outline

Propiedades de $\texttt{follow}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

$$\texttt{follow}_k(X) \; = \; \{w \mid S \overset{\star}{\Rightarrow} \alpha X \beta \text{ y } w \in \texttt{first}_k(\beta\#)\}$$

Si consideramos $X \neq S$:

$$\begin{aligned}
\texttt{follow}_k(X) &= \bigcup_{S \overset{\star}{\Rightarrow} \alpha X \beta} \texttt{first}_k(\beta\#) \\
&= \bigcup_{S \overset{\star}{\Rightarrow} \alpha Y \beta \Rightarrow \alpha\alpha' X \beta' \beta} \texttt{first}_k(\beta'\beta\#) \\
&= \bigcup_{Y \to \alpha' X \beta'} \bigcup_{S \overset{\star}{\Rightarrow} \alpha Y \beta} \texttt{first}_k(\beta'\beta\#) \\
&= \bigcup_{Y \to \alpha' X \beta'} \bigcup_{S \overset{\star}{\Rightarrow} \alpha Y \beta} \texttt{first}_k(\beta') \odot_k \texttt{first}_k(\beta\#) \\
&= \bigcup_{Y \to \alpha' X \beta'} \texttt{first}_k(\beta') \odot_k \bigcup_{S \overset{\star}{\Rightarrow} \alpha Y \beta} \texttt{first}_k(\beta\#) \\
&= \bigcup_{Y \to \alpha' X \beta'} \texttt{first}_k(\beta') \odot_k \texttt{follow}_k(Y)
\end{aligned}$$

Propiedades de $\texttt{follow}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

$$\texttt{follow}_k(X) \; = \; \{w \mid S \overset{\star}{\Rightarrow} \alpha X \beta \text{ y } w \in \texttt{first}_k(\beta\#)\}$$

Si consideramos $X \neq S$:

$$\begin{aligned}
\texttt{follow}_k(X) &= \bigcup_{S \overset{\star}{\Rightarrow} \alpha X \beta} \texttt{first}_k(\beta\#) \\
&\vdots \\
&= \bigcup_{Y \to \alpha' X \beta'} \texttt{first}_k(\beta') \odot_k \texttt{follow}_k(Y)
\end{aligned}$$

Si consideramos $X = S$:

$$\begin{aligned}
\texttt{follow}_k(S) &= \{\#\} \cup \bigcup_{S \overset{+}{\Rightarrow} \alpha S \beta} \texttt{first}_k(\beta\#) \\
&= \{\#\} \cup \bigcup_{S \overset{\star}{\Rightarrow} \alpha Y \beta \Rightarrow \alpha \alpha' S \beta' \beta} \texttt{first}_k(\beta' \beta \#) \\
&= \{\#\} \cup \bigcup_{Y \to \alpha' S \beta'} \texttt{first}_k(\beta') \odot_k \texttt{follow}_k(Y)
\end{aligned}$$

Propiedades de $\texttt{follow}_k$

Sea $\mathcal{G} = (V, \Sigma, P, S)$ una gramática libre de contexto y $k \geq 1$.

Teorema

Para $X \neq S$: $$\texttt{follow}_k(X) = \bigcup_{Y \to \alpha X \beta} \texttt{first}_k(\beta) \odot_k \texttt{follow}_k(Y)$$

$$\texttt{follow}_k(S) = \{\#\} \cup \bigcup_{Y \to \alpha S \beta} \texttt{first}_k(\beta) \odot_k \texttt{follow}_k(Y)$$

Defina el siguiente **programa recursivo** para todo $X \in V$:

Para $X \neq S$: $$\texttt{follow}_k^0(X) := \varnothing$$

$$\texttt{follow}_k^0(S) := \{\#\}$$

Para $X \neq S$: $$\texttt{follow}_k^i(X) := \bigcup_{Y \to \alpha X \beta} \texttt{first}_k(\beta) \odot_k \texttt{follow}_k^{i-1}(Y)$$

$$\texttt{follow}_k^i(S) := \{\#\} \cup \bigcup_{Y \to \alpha S \beta} \texttt{first}_k(\beta) \odot_k \texttt{follow}_k^{i-1}(Y)$$

¿cómo calcular $\texttt{follow}_k$?

Similar al caso de $\texttt{first}_k$:

- $\texttt{follow}_k^{i-1}(X) \subseteq \texttt{follow}_k^{i}(X)$ para todo $i > 1$.
- Como $\texttt{follow}_k(X) \subseteq \Sigma^{\leq k}$, entonces para algún $i \leq k \cdot |\Sigma|^k \cdot |V|$:

$$\texttt{follow}_k^{j}(X) \;=\; \texttt{follow}_k^{j+1}(X) \quad \text{para todo } j \geq i.$$

Teorema

Sea i^* el menor número tal que $\texttt{follow}_k^{i^*}(X) = \texttt{follow}_k^{i^*+1}(X)$ para todo $X \in V$. Entonces para todo $X \in V$:

$$\texttt{follow}_k^{i^*}(X) \;=\; \texttt{follow}_k(X)$$

Demostración: ejercicio.

…y podemos calcular $\texttt{follow}_k(X)$
con un algoritmo similar que $\texttt{first}_k(X)$.

¿cómo calculamos $\texttt{first}_k$ y $\texttt{follow}_k$ eficientemente?

- Algoritmos toman $\mathcal{O}(k \cdot |\Sigma|^k \cdot |V|)$ repeticiones en el peor caso.
- Si $k = 1$, el número de repeticiones será $\mathcal{O}(|\Sigma| \cdot |V|)$ y tiempo del algoritmo será polinomial en $|\mathcal{G}|$ en el peor caso.
- Para $k = 1$ incluso se puede hacer en tiempo $\mathcal{O}(|V| \cdot |P|)$ en total.