Universidad de Santiago de Chile

Facultad de Ingeniería Informática

Teoría de la Computación

Profesora Consuelo Ramirez

PEP 3
Lenguajes Independientes del Contexto

Nicolás Aguilera

Junio 2023

Índice

1. Gramáticas Independientes del Contexto (GIC)

1.1. Simplificación

1. **Anulables:** Se agregan a N_ε todos los símbolos que produzcan ε de manera directa o indirecta.

2. **Eliminación de producciones vacías:** Construcción de una nueva gramática P' donde se agregan las combinaciones de las producciones que contengan símbolos anulables, considerando que un símbolo X pueda ser ε.

3. **Eliminación de producciones unitarias:** Se construye P'' donde se reemplaza cada símbolo unitario con sus producciones.

4. **Eliminación de símbolos inútiles:** Se crea una lista $N_A = \varnothing$ y una lista N' que contenga los símbolos que produzcan palabras. Se actualiza $N_A = N_A \cup N'$ y se crea una nueva lista N'' que agrega símbolos que produzcan palabras a través de los símbolos en N_A. El algoritmo se repite hasta que $N_A = N^n$.

Ejemplo 1:

$$G = (\{S, A, B\}, \{a, b\}, P, S)$$

$$\begin{aligned} P = \{ & \\ & S \to AB \\ & A \to aAA|\varepsilon \\ & B \to bBB|\varepsilon \\ \} & \end{aligned}$$

Solución: Se comienza construyendo la lista de símbolos anulables $N_\varepsilon = \{A, B, S\}$ ya que se puede llegar a ε a través de estos de forma directa o indirecta. Se construye la nueva gramática G':

$$G' = (\{S, A, B\}, \{a, b\}, P', S)$$

$$\begin{aligned} P' = \{ & \\ & S \to AB|A|B \\ & A \to aAA|aA|a \\ & B \to bBB|bB|b \\ \} & \end{aligned}$$

Se construye una nueva gramática eliminando las producciones unitarias:

$$G'' = (\{S, A, B\}, \{a, b\}, P'', S)$$

$$
\begin{aligned}
P'' = \{ & \\
& S \to AB|aAA|aA|a|bBB|bB|b \\
& A \to aAA|aA|a \\
& B \to bBB|bB|b \\
\} &
\end{aligned}
$$

Ejemplo 2: En este ejemplo, a diferencia del Ejemplo 1, se verá el algoritmo del paso número 4, para la eliminación de símbolos inútiles.

$$G = (\{S, A, B, C, D\}, \{a, b, c\}, P, S)$$

$$
\begin{aligned}
P = \{ & \\
& S \to aAAA \\
& A \to aAb|aC \\
& B \to BD|Ac \\
& C \to b \\
\} &
\end{aligned}
$$

Iteración 1:

$$N_A = \varnothing$$
$$N' = \{C\}$$

Iteración 2:

$$N_A = N_A \cup N' = \varnothing \cup \{C\} = \{C\}$$
$$N'' = \{C, A\}$$

Iteración 3:

$$N_A = N_A \cup N'' = \{C\} \cup \{C, A\} = \{C, A\}$$
$$N''' = \{C, A, S, B\}$$

Iteración 4:

$$N_A = N_A \cup N''' = \{C, A\} \cup \{C, A, S, B\} = \{C, A, S, B\}$$
$$N''' = \{C, A, S, B\}$$

Como $N_A = N'''$ se detiene el algoritmo y se construye la nueva gramática:

$$G' = (\{S, A, B, C\}, \{a, b, c\}, P', S)$$

$$
\begin{aligned}
P = \{ & \\
& S \to aAAA \\
& A \to aAb|aC \\
& B \to Ac \\
& C \to b \\
\}&
\end{aligned}
$$

A continuación deben eliminarse todos los símbolos terminales y no terminales que no puedan ser alcanzados desde el símbolo inicial. Quedando como resultado final la siguiente gramática:

$$G'' = (\{S, A, C\}, \{a, b\}, P'', S)$$

$$
\begin{aligned}
P = \{ & \\
& S \to aAAA \\
& A \to aAb|aC \\
& C \to b \\
\}&
\end{aligned}
$$

1.2. Forma Normal de Chomsky

$$A \to BC$$

$$A \to \sigma$$

Se realiza la simplificación vista anteriormente y luego se normaliza. Siguiendo con el ejemplo anterior, las nuevas producciones quedan de la siguiente manera:

$$
\begin{aligned}
P'' = \{ & \\
& S \to aAAA \\
& A \to aAb|aC \\
& C \to b \\
\}&
\end{aligned}
$$

$$
\begin{aligned}
P''' = \{ & \\
& S \to C_aD \\
& C_a \to a \\
& A \to C_aB|C_aC \\
& B \to AC_b \\
& C_b \to b \\
& D \to AE \\
& E \to AA \\
& C \to b \\
\}&
\end{aligned}
$$

2. Autómatas Apiladores

Los autómatas apiladores son autómatas que introducen una pila (o *stack*) y se ven de la siguiente manera:

$$A = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

En este caso Q representa los estados, Σ el alfabeto de entrada, Γ el alfabeto de la pila, δ la función de transición, q_0 el estado inicial, Z_0 el simbolo inicial de la pila y F el conjunto de estados finales.

2.1. Lenguaje Aceptado mediante Estado Final (LA)

El procedimiento para resolver los LA es realizar la traza utilizando las transiciones, partiendo siempre por ε, y luego con los símbolos del alfabeto. Si se llega a un estado final al final de la palabra, entonces se cumple que es LA. Los LA pueden verse de la siguiente manera:

$$A = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \qquad F \neq \varnothing$$

Ejemplo:

$$A = (\{q_0, q_1, q_2\}, \{0, 1, c\}, \{0, 1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

2.2. Lenguaje Aceptado mediante Agotamiento de Pila

El procedimiento para resolver los *NA* es realizar la traza utilizando las transiciones, partiendo siempre por ε y una vez que se agotado la pila y no quedan símbolos por revisar, se dice que el lenguaje es *NA*. Pueden verse de la siguiente forma:

$$A = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \varnothing)$$

Ejemplo:

$$A = (\{q_1, q_2\}, \{0, 1\}, \{R, G, B\}, \delta, q_1, R, \varnothing)$$

Las funciones de transición δ son las siguientes:

$$\begin{aligned}
\delta(q_1, 0, R) &= \{(q_1, BR)\} \\
\delta(q_1, 1, R) &= \{(q_1, GR)\} \\
\delta(q_1, 0, B) &= \{(q_1, BB), (q_2, \varepsilon)\} \\
\delta(q_1, 0, G) &= \{(q_1, BG)\} \\
\delta(q_1, 1, B) &= \{(q_1, GB)\} \\
\delta(q_1, 1, G) &= \{(q_1, GG), (q_2, \varepsilon)\} \\
\delta(q_2, 0, B) &= \{(q_2, \varepsilon)\} \\
\delta(q_2, 1, G) &= \{(q_2, \varepsilon)\} \\
\delta(q_1, \varepsilon, R) &= \{(q_2, \varepsilon)\} \\
\delta(q_2, \varepsilon, R) &= \{(q_2, \varepsilon)\}
\end{aligned}$$

2.3. Equivalencias

2.3.1. LA → NA

En este caso queremos pasar de un estado final a un agotamiento por pila.

$$A = (\{q_1, q_2, q_3\}, \{a, b\}, \{a, z\}, \delta, q_1, z, \{q_3\})$$

$$\begin{aligned}
\delta(q_1, a, z) &= \{(q_1, az)\} \\
\delta(q_1, b, z) &= \{(q_2, \varepsilon)\} \\
\delta(q_1, a, a) &= \{(q_3, a)\}
\end{aligned}$$

1. Se agrega un **nuevo estado inicial** q_1', un **nuevo estado auxiliar** q_e y un **nuevo tope de pila** X_o.

$$A' = (\{q_1', q_e, q_1, q_2, q_3\}, \{a, b\}, \{X_o, a, z\}, \delta', q_1', X_o, \varnothing)$$

2. La transición desde el nuevo estado inicial a través de ε con el nuevo símbolo inicial de la pila X_o es el ex-estado inicial con el ex-símbolo inicial z concatenado con el nuevo símbolo inicial zX_o.

$$\delta'(q_1', \varepsilon, X_o) = \{(q_1, zX_o)\}$$

3. Se mantienen las transiciones originales cambiando a δ' y luego para los ex estados finales se hacen nuevas transiciones con cada símbolo de la pila y ε.

$$\begin{aligned}\delta'(q_3, \varepsilon, a) &= \{(q_e, \varepsilon)\}\\ \delta'(q_3, \varepsilon, z) &= \{(q_e, \varepsilon)\}\\ \delta'(q_3, \varepsilon, X_o) &= \{(q_e, \varepsilon)\}\end{aligned}$$

4. Lo mismo para el nuevo estado auxiliar q_e, también llegando a q_e.

$$\begin{aligned}\delta'(q_e, \varepsilon, a) &= \{(q_e, \varepsilon)\}\\ \delta'(q_e, \varepsilon, z) &= \{(q_e, \varepsilon)\}\\ \delta'(q_e, \varepsilon, X_o) &= \{(q_e, \varepsilon)\}\end{aligned}$$

2.3.2. NA $\rightarrow$ LA

En este caso queremos realizar la transformación inversa con el mismo ejemplo anterior, pero en este caso no se tiene estado final.

$$A = (\{q_1, q_2, q_3\}, \{a, b\}, \{a, z\}, \delta, q_1, z, \varnothing)$$

$$\begin{aligned}\delta(q_1, a, z) &= \{(q_1, az)\}\\ \delta(q_1, b, z) &= \{(q_2, \varepsilon)\}\\ \delta(q_1, a, a) &= \{(q_3, a)\}\end{aligned}$$

1. Se agrega un **nuevo estado inicial** q_1', un **nuevo estado final** q_F y un **nuevo tope de pila** X_o.

$$A' = (\{q_1', q_F, q_1, q_2, q_3\}, \{a, b\}, \{X_o, a, z\}, \delta', q_1', X_o, \{q_F\})$$

2. La transición desde el nuevo estado inicial a través de ε con el nuevo símbolo inicial de la pila X_o es el ex-estado inicial con el ex-símbolo inicial z concatenado con el nuevo símbolo inicial zX_o.

$$\delta'(q_1', \varepsilon, X_o) = \{(q_1, zX_o)\}$$

3. Se copian las transiciones originales y para los estados originales se hacen transiciones a través de ε con X_o, llegando al estado final con ε.

$$\begin{aligned}\delta(q_1, \varepsilon, X_o) &= \{(q_F, \varepsilon)\}\\ \delta(q_2, \varepsilon, X_o) &= \{(q_F, \varepsilon)\}\\ \delta(q_3, \varepsilon, X_o) &= \{(q_F, \varepsilon)\}\end{aligned}$$

2.3.3. GIC → AA

$$G = (\{A, B, C\}, \{a, b, c\}, P, A)$$

$$\begin{aligned} P = \{ & \\ & A \to abB|\varepsilon \\ & B \to ac|Ca \\ & C \to c|\varepsilon \\ \} & \end{aligned}$$

1. Se crea un **nuevo estado inicial**, se mantiene el alfabeto original y para los símbolos del *stack* se hace la unión entre los símbolos **terminales y no terminales**, se deja el ex estado inicial del GIC como primer símbolo de la pila y el estado final será vacío.

$$A = (\{q_0\}, \{a, b, c\}, \{A, B, C, a, b, c\}, \delta, q_0, A, \varnothing)$$

2. Se hace la transición del nuevo estado con los **ex no terminales** a través de ε, llegando a los estados iniciales con las producciones de cada símbolo.

$$\begin{aligned} \delta(q_0, \varepsilon, A) =& \{(q_0, abB), (q_0, \varepsilon)\} \\ \delta(q_0, \varepsilon, B) =& \{(q_0, ac), (q_0, Ca)\} \\ \delta(q_0, \varepsilon, C) =& \{(q_0, c), (q_0, \varepsilon)\} \end{aligned}$$

2.4. Primero y Siguiente

Para una gramática $G = (N, \Sigma, P, S)$ se presentan los conceptos de primero y siguiente que serán usados para la construcción de la *tabla sintáctica*.

2.4.1. Primero

A continuación se presentan las reglas para el cálculo de los primeros.

$$\begin{aligned} & P(\varepsilon) = \varnothing \\ & P(\sigma\alpha) = \{\sigma\} \\ & A \to \alpha_1|\alpha_2|...|\alpha_n \in P \quad \Rightarrow \quad P(A) = P(\alpha_1) \cup P(\alpha_2) \cup ... \cup P(\alpha_n) \\ & \alpha = X_1X_2...X_n \end{aligned}$$

En este caso σ representa un símbolo no terminal y α las producciones de símbolos no terminales.

2.4.2. Siguiente

A continuación se presentan las reglas para el cálculo de los siguientes.

- Los *siguientes* de todos los no terminales son vacíos en un comienzo a excepción del símbolo inicial de una gramática que comienza con {$}.
- Si $A \to \alpha B\beta \quad \Rightarrow \quad S(B) = P(\beta)$
- Si $A \to \alpha B \quad \Rightarrow \quad S(B) = S(A)$
- Si $A \to \alpha B\beta \quad \wedge \quad \beta \in N_\varepsilon \quad \Rightarrow \quad S(B) = P(\beta) \cup S(A)$

2.5. Eliminación de la recursividad por la izquierda

$$\begin{aligned} A \to Aa|\beta \in P \quad \Rightarrow \quad & A \to \beta A' \\ & A' \to aA'|\varepsilon \end{aligned}$$

Ejemplo:

$$G = (\{E, T, F\}, \{+, *, (,), i\}, P, E)$$

$$\begin{aligned} P = \{ & \\ & E \to E + T|T \\ & T \to T * F|F \\ & F \to (E)|i \\ \} & \end{aligned}$$

1. Por cada recursión se agrega un nuevo no terminal y se cambian las producciones siguiendo la regla descrita anteriormente, obtiendo una nueva gramática G'.

$$G' = (\{E, T, F, E', T'\}, \{+, *, (,), i\}, P', E)$$

$$\begin{aligned} P' = \{ & \\ & E \to TE' \\ & E' \to +TE'|\varepsilon \\ & T \to FT' \\ & T' \to *FT'|\varepsilon \\ & F \to (E)|i \\ \} & \end{aligned}$$

2.6. Tabla de analizador sintáctico (LL)

1. Se realiza la eliminación de la recursión por la izquierda en caso de ser necesario.
2. Se procede con el cálculo de los *primeros* y *siguientes* de cada no terminal.
3. Por último se sigue con la construcción de la tabla.

Siguiendo con el ejemplo anterior visto en eliminación de la recursividad por la izquierda procedemos con el cálculo de los *primeros* y *siguientes* de los no terminales. Hay que tener en cuenta además los símbolos anulables los cuales son $N_\varepsilon = \{E', T'\}$.

$$\begin{aligned} P(E) &= P(TE') \\ &= P(T) \cup \cancel{P(E')} \\ &= P(FT') \\ &= P(F) \cup \cancel{P(T')} \\ &= P((E)) \cup P(i) \\ &= \{(\} \cup \{i\} \\ &= \{(, i\} = P(T) = P(F) \end{aligned}$$

$$\begin{aligned} P(E') &= P(+TE') \cup P(\varepsilon) \\ &= \{+\} \cup \varnothing \\ &= \{+\} \end{aligned}$$

$$\begin{aligned} P(T') &= P(*FT') \cup P(\varepsilon) \\ &= \{*\} \cup \varnothing \\ &= \{*\} \end{aligned}$$

A continuación se realiza el cálculo de los siguientes:

$$\begin{aligned} S(E) &= \{\$\} \cup P()) \\ &= \{\$\} \cup \{)\} \\ &= \{\$,)\} \end{aligned}$$

$$\begin{aligned} S(E') &= S(E) \cup \cancel{S(E')} \\ &= \{\$,)\} \end{aligned}$$

$$\begin{aligned} S(T) &= P(E') \cup S(E) \cup P(E') \cup S(E') \\ &= \{+\} \cup \{\$,)\} \cup \{+\} \cup \{\$,)\} \\ &= \{+, \$,)\} \end{aligned}$$

$$\begin{aligned} S(T') &= S(T) \cup \cancel{S(T')} \\ &= \{+, \$,)\} \end{aligned}$$

$$\begin{aligned} P(F) &= P(T') \cup S(T) \cup P(T') \cup S(T') \\ &= \{*\} \cup \{+, \$,)\} \cup \{*\} \cup \{+, \$,)\} \\ &= \{*, +, \$,)\} \end{aligned}$$

A continuación se construye una tabla donde las filas corresponden a los símbolos no terminales y las columnas a los símbolos tal y como se ve en el Cuadro 1.

El primer paso es llenar con los *primeros* de cada símbolo no terminal. Luego, llenar con los siguientes de los anulables, es decir $\in N_\varepsilon$. Finalmente, se llena con los siguientes de los no terminales. Si está vacío y coincide con el símbolo, se coloca SINC.

	+	*	(	)	i	$
E			E→ TE'	SINC	E → TE'	SINC
E'	E' → + TE'			E'→ ε		E'→ ε
T	SINC		T' → FT'	SINC	T → FT'	SINC
T'	T' → ε	T' → * FT'		T' → ε		T' → ε
F	SINC	SINC	F → (E)	SINC	F → i	SINC

Cuadro 1: Tabla de analizador sintáctico del ejemplo.

2.7. Recuperación de errores en analizador sintáctico LL

Siguiendo con el ejemplo anterior queremos ver si la palabra $\omega = +i * +i$ cumple con las condiciones (?).

Stack	Entrada	Salida
$E\$$	$i + i * i\$$	$E \to TE'$
$TE'\$$	$i + i * i\$$	$T \to FT'$
$FT'E'\$$	$i + i * i\$$	$F \to i$
$iT'E'\$$	$i + i * i\$$	$E \to TE'$
$T'E'\$$	$+i * i\$$	$T' \to \varepsilon$
$E'\$$	$+i * i\$$	$E' \to +TE'$
$+TE'\$$	$+i * i\$$	
$TE'\$$	$i * i\$$	$T \to FT'$
$FT'E'\$$	$i * i\$$	$F \to i$
$iT'E'\$$	$i * i\$$	
$T'E'\$$	$*i\$$	$T' \to *FT'$
$*FT'E'\$$	$*i\$$	
$FT'E'\$$	$i\$$	$F \to i$
$iT'E'\$$	$i\$$	
$T'E'\$$	$\$$	$T' \to \varepsilon$
$E'\$$	$\$$	$E' \to \varepsilon$
$\$$	$\$$	Aceptado

Cuadro 2: Recuperación de errores para LL.

Para verificar si la palabra es aceptada por el LL, se comienza la pila con el símbolo inicial y la entrada a verificar. Se siguen las siguientes instrucciones:

- Si el tope de la pila coincide con el tope de la entrada (ambos no terminales), se eliminan de las pilas y se continua.
- Si el tope de la pila es un símbolo no terminal, se busca en la tabla la producción en la coordenada (X_i, σ) y se reemplaza el tope de la pila por la producción.
- Si el tope de pila y la entrada son ambos $, entonces la palabra es aceptada.

2.8. Analizadores Sintácticos LR

La L representa la lectura de la entrada de izquierda a derecha, la R representa una derivación por la derecha en orden inverso, y el 1 es por utilizar un símbolo de entrada de anticipación en cada paso para tomar las decisiones de la acción en el análisis sintáctico.

2.8.1. Analizador sintáctico SLR

$$G = (\{S, A\}, \{a, b\}, P, S)$$

$$\begin{aligned} P = \{ & \\ & S \to SA|A \\ & A \to aSb|ab \\ \} & \end{aligned}$$

En primer lugar se comienza agregando un nuevo estado auxiliar S' que produzca el ex símbolo inicial, y se enumeran las demás producciones como se muestra abajo:

$$\begin{aligned} P = \{ & \\ & S' \to S \\ & S \to SA^{1}|A^{2} \\ & A \to aSb^{3}|ab^{4} \\ \} & \end{aligned}$$

Luego se comienza con la construcción de estados, siguiendo los siguientes pasos:

1. Se comienza por el estado I_o colocando la primera producción de S' con el símbolo de clausura $\bullet$ al principio de la producción.

2. Si la clausura antecede algún símbolo no terminal, se deben agregar las producciones de dichos símbolos con la clausura al comienzo también.

3. Luego se agrupan en nuevos estados I_k las producciones que su clausura anteceda el mismo símbolo (terminal o no terminal). **Es importante** no crear nuevos estados que ya existan, es decir, si el conjunto de producciones agrupadas por la clausura ya existe anteriormente, se debe agrupar y marcar con el nombre de ese estado existente.

4. Para los nuevos estados, se comienza con las producciones que fueron agrupadas por el mismo símbolo, moviendo la clausura una posición a la derecha. Y se verifica nuevamente el paso 2.

5. Si al mover la clausura, esta no antecede ningún símbolo (es decir, no hay nada a la derecha) la producción es aceptada y se marca.

6. El algoritmo termina cuando no quedan más estados por revisar.

$$
\begin{array}{lll}
I_0: & S' \to \bullet S & \xrightarrow{S} I_1 \\
& S \to \bullet SA & \xrightarrow{S} I_1 \\
& S \to \bullet A & \xrightarrow{A} I_2 \\
& A \to \bullet aSb & \xrightarrow{a} I_3 \\
& A \to \bullet ab & \xrightarrow{a} I_3
\end{array}
$$

$$
\begin{array}{lll}
I_1: & S' \to S\bullet & \\
& S \to S \bullet A & \xrightarrow{A} I_4 \\
& A \to \bullet aSb & \xrightarrow{a} I_3 \\
& A \to \bullet ab & \xrightarrow{a} I_3
\end{array}
$$

$$I_2: \quad S \to A\bullet \quad (\star)$$

$$
\begin{array}{lll}
I_3: & A \to a \bullet Sb & \xrightarrow{S} I_5 \\
& A \to a \bullet b & \xrightarrow{b} I_6 \\
& S \to \bullet SA & \xrightarrow{S} I_5 \\
& S \to \bullet A & \xrightarrow{A} I_2 \\
& A \to \bullet aSb & \xrightarrow{a} I_3 \\
& A \to \bullet ab & \xrightarrow{a} I_3
\end{array}
$$

$$I_4: \quad S \to SA\bullet \quad (\star)$$

$$
\begin{array}{lll}
I_5: & A \to aS \bullet b & \xrightarrow{b} I_7 \\
& S \to S \bullet A & \xrightarrow{A} I_4 \\
& A \to \bullet aSb & \xrightarrow{a} I_3 \\
& A \to \bullet ab & \xrightarrow{a} I_3
\end{array}
$$

$$I_6: \quad A \to ab\bullet \quad (\star)$$

$$I_7: \quad A \to aSb\bullet \quad (\star)$$

Ahora se calcular los *siguientes* de las producciones marcadas con ($\star$):

$$
\begin{aligned}
S(S) &= \{\$\} \cup P(A) \cup P(b) \\
&= \{\$\} \cup P(aSb) \cup P(ab) \cup \{b\} \\
&= \{\$\} \cup \{a\} \cup \{a\} \cup \{b\} \\
&= \{\$, a, b\}
\end{aligned}
\qquad
\begin{aligned}
S(A) &= S(S) \cup P(b) \\
&= \{\$, a, b\}
\end{aligned}
$$

Finalmente se construye una tabla con los estados por filas y los símbolos terminales y no terminales como columnas. Se va revisando por estado, si la agrupación es con un símbolo no terminal, se coloca solo el número del nuevo estado. Si el símbolo es terminal, se coloca D_k, donde k es el número del estado. Finalmente, para las producciones marcadas con $(\star)$, se revisa el número producción marcado al comienzo y se marcan los símbolos con R_p, donde p es el número de la producción del estado finalizado.

	a	b	$	S	A
0	D_3			1	2
1	D_3		A		4
2	R_2	R_2	R_2		
3	D_3	D_6		5	2
4	R_1	R_1	R_1		
5	D_3	D_7			4
6	R_4	R_4	R_4		
7	R_3	R_3	R_3		

3. Bibliografía y referencias

1. Stallings W. (2005). *Sistemas Operativos. Aspectos internos y aspectos de diseño.* (5ta ed.). Pearson.

2. Modelo GPT-3.5, OpenAI (2021). ChatGPT. [Software informático]. Disponible en https://openai.com