Análisis Sintáctico
Parte II

Albert Rubio

Procesadores de Lenguajes, FdI, UCM

Doble Grado Matématicas e Informática

Análisis Sintáctico. Parte II

1. Analizadores ascendentes $LR(k)$

2. Analizadores descendentes $LL(k)$

3. Patrones de especificación

Contenidos

Analizadores ascendentes

Los analizadores ascendentes

- leen símbolos de la entrada de izquierda a derecha
- construyen el árbol sintáctico en sentido **ascendente**
- siguen una derivación por la derecha en sentido inverso

Deben reconocer **partes derechas** de las reglas:

- si detectan un símbolo que continúa lo que hay en la pila aplican un **desplazamiento**: consumiendo símbolo y apilándolo. **Shift**
- si detectan una parte derecha aplican una **reducción**: sustituyendo en la pila la parte derecha por la izquierda. **Reduce**

Ejemplo de reconocimiento

$$S \Longrightarrow_{de} E \Longrightarrow_{de} T \Longrightarrow_{de} T * F \Longrightarrow_{de} T * id \Longrightarrow_{de} F * id \Longrightarrow_{de} id * id$$

$S \longrightarrow E$
$E \longrightarrow E + T \mid T$
$T \longrightarrow T * F \mid F$
$F \longrightarrow$ "(" E ")" $\mid$ **id**

Pila	Entrada	Acción
	id * id ⊢	D
id	* id ⊢	R
F	* id ⊢	R
T	* id ⊢	D
T *	id ⊢	D
T * id	⊢	R
T * F	⊢	R
T	⊢	R
E	⊢	R
S	⊢	

Asidero (handle)

$$S \Longrightarrow_{de} E \Longrightarrow_{de} T \Longrightarrow_{de} T * F \Longrightarrow_{de} T * id \Longrightarrow_{de} F * id \Longrightarrow_{de} id * id$$

En sentido inverso: en cada paso se reemplaza α por A para alguna regla $A \longrightarrow \alpha$.

Llamamos **asidero** de la forma de frase derecha (ffd) al α que hay que reducir en cada paso de la derivación por la derecha en sentido inverso. Ejemplos:

-

Asidero (handle)

$$S \Longrightarrow_{de} E \Longrightarrow_{de} T \Longrightarrow_{de} T * F \Longrightarrow_{de} T * id \Longrightarrow_{de} F * id \Longrightarrow_{de} id * id$$

En sentido inverso: en cada paso se reemplaza α por A para alguna regla $A \longrightarrow \alpha$.

Llamamos **asidero** de la forma de frase derecha (ffd) al α que hay que reducir en cada paso de la derivación por la derecha en sentido inverso. Ejemplos:

- En $T * id$ el asidero es

Asidero (handle)

$$S \Longrightarrow_{de} E \Longrightarrow_{de} T \Longrightarrow_{de} T * F \Longrightarrow_{de} T * id \Longrightarrow_{de} F * id \Longrightarrow_{de} id * id$$

En sentido inverso: en cada paso se reemplaza α por A para alguna regla $A \longrightarrow \alpha$.

Llamamos **asidero** de la forma de frase derecha (ffd) al α que hay que reducir en cada paso de la derivación por la derecha en sentido inverso. Ejemplos:

- En $T * id$ el asidero es id

Asidero (handle)

$$S \Longrightarrow_{de} E \Longrightarrow_{de} T \Longrightarrow_{de} T * F \Longrightarrow_{de} T * id \Longrightarrow_{de} F * id \Longrightarrow_{de} id * id$$

En sentido inverso: en cada paso se reemplaza α por A para alguna regla $A \longrightarrow \alpha$.

Llamamos **asidero** de la forma de frase derecha (ffd) al α que hay que reducir en cada paso de la derivación por la derecha en sentido inverso. Ejemplos:

- En $T * id$ el asidero es id
- En $F * id$ el asidero es

Asidero (handle)

$$S \Longrightarrow_{de} E \Longrightarrow_{de} T \Longrightarrow_{de} T * F \Longrightarrow_{de} T * id \Longrightarrow_{de} F * id \Longrightarrow_{de} id * id$$

En sentido inverso: en cada paso se reemplaza α por A para alguna regla $A \longrightarrow \alpha$.

Llamamos **asidero** de la forma de frase derecha (ffd) al α que hay que reducir en cada paso de la derivación por la derecha en sentido inverso.
Ejemplos:

- En $T * id$ el asidero es id
- En $F * id$ el asidero es F

Asidero (handle)

$$S \Longrightarrow_{de} E \Longrightarrow_{de} T \Longrightarrow_{de} T * F \Longrightarrow_{de} T * id \Longrightarrow_{de} F * id \Longrightarrow_{de} id * id$$

En sentido inverso: en cada paso se reemplaza α por A para alguna regla $A \longrightarrow \alpha$.

Llamamos **asidero** de la forma de frase derecha (ffd) al α que hay que reducir en cada paso de la derivación por la derecha en sentido inverso. Ejemplos:

- En $T * id$ el asidero es id
- En $F * id$ el asidero es F

En una gramática no ambigua, el asidero es **único**.

Prefijos viables

$$E \Longrightarrow_{de} E + T \Longrightarrow_{de} E + F \Longrightarrow_{de} E + \mathbf{id} \Longrightarrow_{de} T + \mathbf{id} \Longrightarrow_{de}$$
$$T * F + \mathbf{id} \Longrightarrow_{de} T * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} F * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} \mathbf{id} * \mathbf{id} + \mathbf{id}$$

Dada una ffd $\beta\alpha u$ con ($u \in V_T^*$ y) asidero α
su conjunto de **prefijos viables** son todos los prefijos de $\beta\alpha$

Para la ffd $T * \mathbf{id} + \mathbf{id}$ tenemos que

Prefijos viables

$E \Longrightarrow_{de} E + T \Longrightarrow_{de} E + F \Longrightarrow_{de} E + \mathbf{id} \Longrightarrow_{de} T + \mathbf{id} \Longrightarrow_{de}$
$T * F + \mathbf{id} \Longrightarrow_{de} T * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} F * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} \mathbf{id} * \mathbf{id} + \mathbf{id}$

Dada una ffd $\beta\alpha u$ con ($u \in V_T^*$ y) asidero α
su conjunto de **prefijos viables** son todos los prefijos de $\beta\alpha$

Para la ffd $T * \underline{\mathbf{id}} + \mathbf{id}$ tenemos que

- el asidero es el **id** más a la izquierda.

Prefijos viables

$$E \Longrightarrow_{de} E + T \Longrightarrow_{de} E + F \Longrightarrow_{de} E + \mathbf{id} \Longrightarrow_{de} T + \mathbf{id} \Longrightarrow_{de}$$
$$T * F + \mathbf{id} \Longrightarrow_{de} T * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} F * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} \mathbf{id} * \mathbf{id} + \mathbf{id}$$

Dada una ffd $\beta\alpha u$ con ($u \in V_T^*$ y) asidero α
su conjunto de **prefijos viables** son todos los prefijos de $\beta\alpha$

Para la ffd $T * \underline{\mathbf{id}} + \mathbf{id}$ tenemos que

- el asidero es el **id** más a la izquierda.
- los prefijos viables son $T * \mathbf{id}$, $T*$, T

Prefijos viables

$$E \Longrightarrow_{de} E + T \Longrightarrow_{de} E + F \Longrightarrow_{de} E + \mathbf{id} \Longrightarrow_{de} T + \mathbf{id} \Longrightarrow_{de}$$
$$T * F + \mathbf{id} \Longrightarrow_{de} T * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} F * \mathbf{id} + \mathbf{id} \Longrightarrow_{de} \mathbf{id} * \mathbf{id} + \mathbf{id}$$

Dada una ffd $\beta\alpha u$ con ($u \in V_T^*$ y) asidero α
su conjunto de **prefijos viables** son todos los prefijos de $\beta\alpha$

Para la ffd $T * \underline{\mathbf{id}} + \mathbf{id}$ tenemos que

- el asidero es el **id** más a la izquierda.
- los prefijos viables son $T * \mathbf{id}$, $T*$, T

El trabajo del analizador ascendente consiste en reconocer prefijos viables:

- mientras no ha llegado al asidero, el autómata procede por **desplazamiento**.
- cuando llega al asidero del prefijo viable procede por **reducción**.

Reconocimiento de prefijos viables

El lenguaje de prefijos viables es regular.

El AFD que reconoce este lenguaje puede ser usado para reconocer asideros en un analizador LR.

El autómata característico del autómata de items $car(K_G)$ es un AFN reconoce prefijos viables o asideros según qué estados consideremos finales:

$$car(K_G) = (It_G, V_T \cup V_N, \Delta_c, [S_0 \longrightarrow \bullet S], F)$$

Reconocimiento de prefijos viables

El lenguaje de prefijos viables es regular.

El AFD que reconoce este lenguaje puede ser usado para reconocer asideros en un analizador LR.

El autómata característico del autómata de items $car(K_G)$ es un AFN reconoce prefijos viables o asideros según qué estados consideremos finales:

$$car(K_G) = (It_G, V_T \cup V_N, \Delta_c, [S_0 \longrightarrow \bullet S], F)$$

- desplazamiento en K_G

Reconocimiento de prefijos viables

El lenguaje de prefijos viables es regular.

El AFD que reconoce este lenguaje puede ser usado para reconocer asideros en un analizador LR.

El autómata característico del autómata de items $car(K_G)$ es un AFN reconoce prefijos viables o asideros según qué estados consideremos finales:

$$car(K_G) = (It_G, V_T \cup V_N, \Delta_c, [S_0 \longrightarrow \bullet S], F)$$

- $\Delta_c([X \longrightarrow \alpha \bullet M\beta], M, [X \longrightarrow \alpha M \bullet \beta]) \quad \forall\, X \longrightarrow \alpha M\beta \in P$

Reconocimiento de prefijos viables

El lenguaje de prefijos viables es regular.

El AFD que reconoce este lenguaje puede ser usado para reconocer asideros en un analizador LR.

El autómata característico del autómata de items $car(K_G)$ es un AFN reconoce prefijos viables o asideros según qué estados consideremos finales:

$$car(K_G) = (It_G, V_T \cup V_N, \Delta_c, [S_0 \longrightarrow \bullet S], F)$$

- $\Delta_c([X \longrightarrow \alpha \bullet M\beta], M, [X \longrightarrow \alpha M \bullet \beta]) \quad \forall\, X \longrightarrow \alpha M\beta \in P$

  expansión en K_G

Reconocimiento de prefijos viables

El lenguaje de prefijos viables es regular.

El AFD que reconoce este lenguaje puede ser usado para reconocer asideros en un analizador LR.

El autómata característico del autómata de items $car(K_G)$ es un AFN reconoce prefijos viables o asideros según qué estados consideremos finales:

$$car(K_G) = (It_G, V_T \cup V_N, \Delta_c, [S_0 \longrightarrow \bullet S], F)$$

- $\Delta_c([X \longrightarrow \alpha \bullet M\beta], M, [X \longrightarrow \alpha M \bullet \beta]) \quad \forall\, X \longrightarrow \alpha M\beta \in P$

  $\Delta_c([X \longrightarrow \alpha \bullet Y\beta], \epsilon, [Y \longrightarrow \bullet\gamma]) \quad \forall\, X \longrightarrow \alpha Y\beta,\; Y \longrightarrow \gamma \in P$

Reconocimiento de prefijos viables

El lenguaje de prefijos viables es regular.

El AFD que reconoce este lenguaje puede ser usado para reconocer asideros en un analizador LR.

El autómata característico del autómata de items $car(K_G)$ es un AFN reconoce prefijos viables o asideros según qué estados consideremos finales:

$$car(K_G) = (It_G, V_T \cup V_N, \Delta_c, [S_0 \longrightarrow \bullet S], F)$$

- $\Delta_c([X \longrightarrow \alpha \bullet M\beta], M, [X \longrightarrow \alpha M \bullet \beta]) \quad \forall\, X \longrightarrow \alpha M\beta \in P$

  $\Delta_c([X \longrightarrow \alpha \bullet Y\beta], \epsilon, [Y \longrightarrow \bullet\gamma]) \quad \forall\, X \longrightarrow \alpha Y\beta,\ Y \longrightarrow \gamma \in P$

  reducción en K_G no se incluye. Se aplicaría en los estados $[X \longrightarrow \alpha\bullet]$.

Reconocimiento de prefijos viables

El lenguaje de prefijos viables es regular.

El AFD que reconoce este lenguaje puede ser usado para reconocer asideros en un analizador LR.

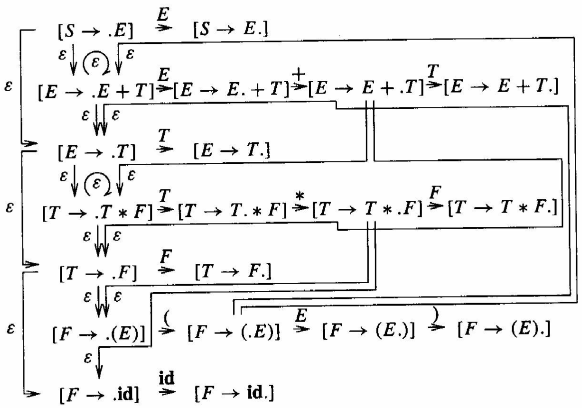
El autómata característico del autómata de items $car(K_G)$ es un AFN reconoce prefijos viables o asideros según qué estados consideremos finales:

$$car(K_G) = (It_G, V_T \cup V_N, \Delta_c, [S_0 \longrightarrow \bullet S], F)$$

- $\Delta_c([X \longrightarrow \alpha \bullet M\beta], M, [X \longrightarrow \alpha M \bullet \beta]) \quad \forall\, X \longrightarrow \alpha M\beta \in P$
  $\Delta_c([X \longrightarrow \alpha \bullet Y\beta], \epsilon, [Y \longrightarrow \bullet\gamma]) \quad \forall\, X \longrightarrow \alpha Y\beta,\ Y \longrightarrow \gamma \in P$
- Si $F = \{[X \longrightarrow \alpha\bullet] \mid X \longrightarrow \alpha \in P\}$ reconoce asideros.
  Si $F = It_G$ reconoce prefijos viables.

$car(K_G)$

$S \longrightarrow E$
$E \longrightarrow E + T \mid T$
$T \longrightarrow T * F \mid F$
$F \longrightarrow "(" \ E \ ")" \mid \mathbf{id}$



Items válidos

$[X \longrightarrow \alpha \bullet \beta]$ es válido para el prefijo viable $\gamma\alpha$ si existe

$$S \Longrightarrow^*_{de} \gamma X u \Longrightarrow_{de} \gamma\alpha\beta u$$

Teorema:
$([S' \longrightarrow \bullet S], \psi) \vdash^*_{car(K_G)} (q, \epsilon)$ si y solo si q es un ítem válido para ψ con q de la forma $[X \longrightarrow \alpha \bullet \beta]$ y $\psi = \gamma\alpha$.

Si $q \in F$ entonces $\beta = \epsilon$ y α es un asidero.

Notad que se asume que la gramática está reducida (en particular que no hay improductivos).

El indeterminismo de $car(K_G)$ implica que hay varios items válidos $[X \longrightarrow \alpha \bullet \beta]$ para un mismo ψ, que representan distintos análisis.

Si determinizamos $car(K_G)$ todos los items válidos para ψ estarán en el mismo estado.

$AFD - LR(0)_G$

El AFD equivalente a $car(K_G)$ se denomina $AFD - LR(0)_G$

- sus estados son conjuntos de items.
- el estado inicial contiene el ítem $[S' \longrightarrow \bullet S]$.
- los estados finales contienen algún ítem $[X \longrightarrow \alpha\bullet]$.

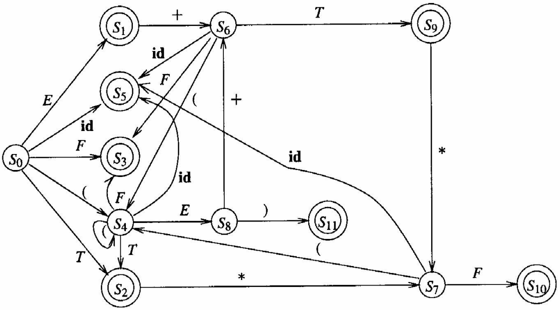
El lenguaje regular reconocido por $AFD - LR(0)_G$ es el de los prefijos viables de G que terminan en un **asidero**.

Si todos los estados fueran finales, el lenguaje sería el de los prefijos viables de G

Camino seguido para construir $AFD - LR(0)_G$:
$G \rightarrow K_G \rightarrow car(K_G) \rightarrow AFD - LR(0)_G$

Ejemplo $AFD - LR(0)_G$

$$
\begin{aligned}
S &\longrightarrow E \\
E &\longrightarrow E + T \mid T \\
T &\longrightarrow T * F \mid F \\
F &\longrightarrow "(" \; E \; ")" \mid \mathbf{id}
\end{aligned}
$$



$S_0 = \{ [S \to .E], [E \to .E + T], [E \to .T], [T \to .T * F], [T \to .F], [F \to .(E)], [F \to .\mathbf{id}]\}$

$S_1 = \{ [S \to E.], [E \to E. + T]\}$

$S_2 = \{ [E \to T.], [T \to T. * F]\}$

$S_3 = \{ [T \to F.]\}$

$S_4 = \{ [F \to (.E)], [E \to .E + T], [E \to .T], [T \to .T * F] [T \to .F] [F \to .(E)] [F \to .\mathbf{id}]\}$

$S_5 = \{ [F \to \mathbf{id}.]\}$

$S_6 = \{ [E \to E + .T], [T \to .T * F], [T \to .F], [F \to .(E)], [F \to .\mathbf{id}]\}$

$S_7 = \{ [T \to T * .F], [F \to .(E)], [F \to .\mathbf{id}]\}$

$S_8 = \{ [F \to (E.)], [E \to E. + T]\}$

$S_9 = \{ [E \to E + T.], [T \to T. * F]\}$

$S_{10} = \{ [T \to T * F.]\}$

$S_{11} = \{ [F \to (E).]\}$

Autómata $LR(0)$

El autómata $AFD - LR(0)_G$ se puede asociar a un autómata con pila $(V_T, Q, \Delta, q_0, \{q_f\})$ en el que los estados de Q son conjuntos de items.

Este autómata se llama autómata $LR(0)$ de G.

Permite reconocer G eficientemente si es no ambigua con lookahead 0.

Los estados de $AFD - LR(0)$ son muy informativos sobre el tipo de transiciones que puede hacer el autómata con pila LR(0):

- Si contiene $[X \longrightarrow \alpha \bullet a\beta]$ existe una transición de desplazamiento
- Si contiene $[X \longrightarrow \alpha\bullet]$ existe una transición de reducción.

Estados inadecuados

En las siguientes situaciones tenemos un conflicto:

- Si existen dos items $[X \longrightarrow \alpha \bullet a\beta]$ y $[Y \longrightarrow \gamma\bullet]$ en un mismo q tenemos un **conflicto desplazamiento-reducción** (shift-reduce).
- Si existen dos items $[X \longrightarrow \alpha\bullet]$ y $[Y \longrightarrow \gamma\bullet]$ en un mismo q tenemos un **conflicto reducción-reducción** (reduce-reduce).

En ambos casos los estados se llaman **inadecuados**.

Si en $AFD - LR(0)$ no hay estados inadecuados entonces el autómata con pila $LR(0)$ es determinista.

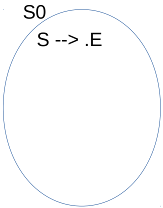
Estados inadecuados

S_1, S_2 y S_9 son inadecuados.

$$
\begin{aligned}
S_0 = \{\ & [S \to .E], [E \to .E + T], [E \to .T], [T \to .T * F], [T \to .F], [F \to .(E)], [F \to .\mathbf{id}]\} \\
S_1 = \{\ & [S \to E.], [E \to E. + T]\} \\
S_2 = \{\ & [E \to T.], [T \to T. * F]\} \\
S_3 = \{\ & [T \to F.]\} \\
S_4 = \{\ & [F \to (.E)], [E \to .E + T], [E \to .T], [T \to .T * F] [T \to .F] [F \to .(E)] [F \to .\mathbf{id}]\} \\
S_5 = \{\ & [F \to \mathbf{id}.]\} \\
S_6 = \{\ & [E \to E + .T], [T \to .T * F], [T \to .F], [F \to .(E)], [F \to .\mathbf{id}]\} \\
S_7 = \{\ & [T \to T * .F], [F \to .(E)], [F \to .\mathbf{id}]\} \\
S_8 = \{\ & [F \to (E.)], [E \to E. + T]\} \\
S_9 = \{\ & [E \to E + T.], [T \to T. * F]\} \\
S_{10} = \{\ & [T \to T * F.]\} \\
S_{11} = \{\ & [F \to (E).]\}
\end{aligned}
$$

Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id

S0

S --> .E
E --> .E + T
E --> .T

Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id

S0

S --> .E
E --> .E + T
E --> .T
T --> .T * F
T --> .F

Cálculo directo de $AFD - LR(0)_G$

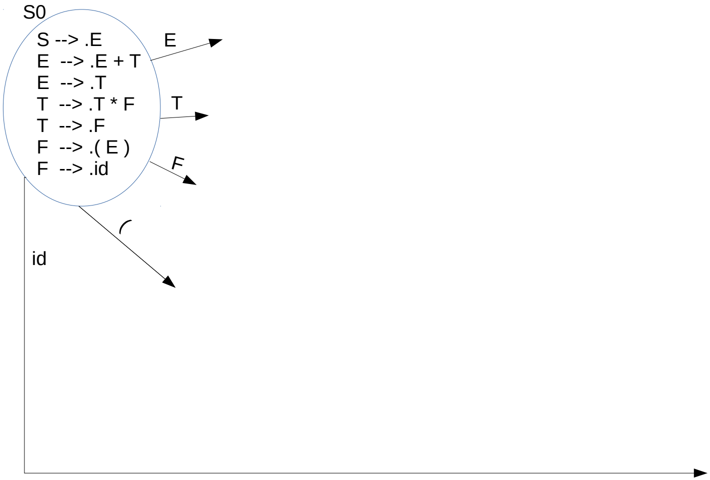
S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id

S0

S --> .E
E --> .E + T
E --> .T
T --> .T * F
T --> .F
F --> .(E)
F --> .id

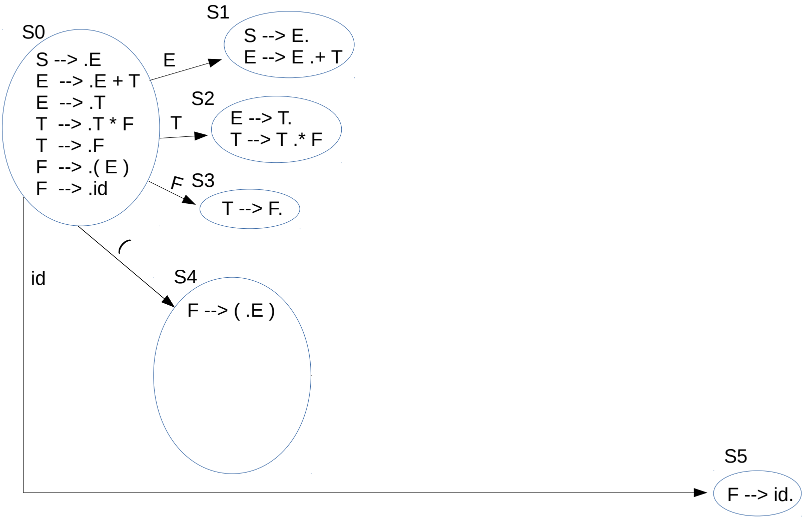
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



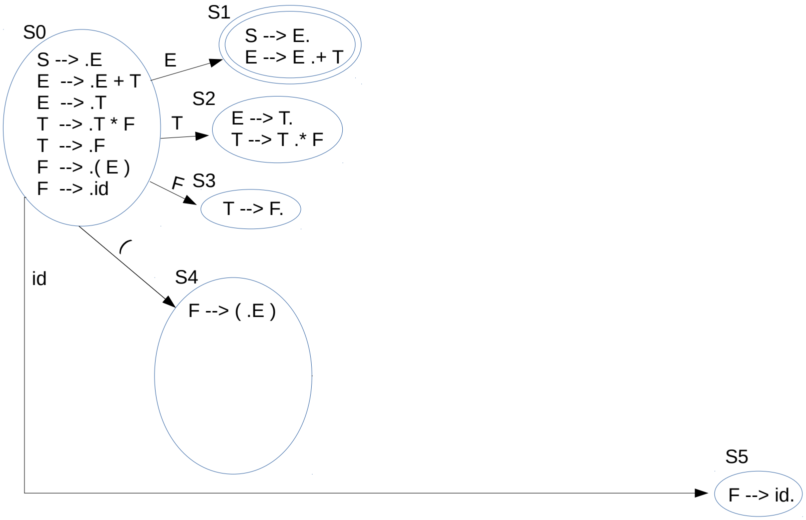
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



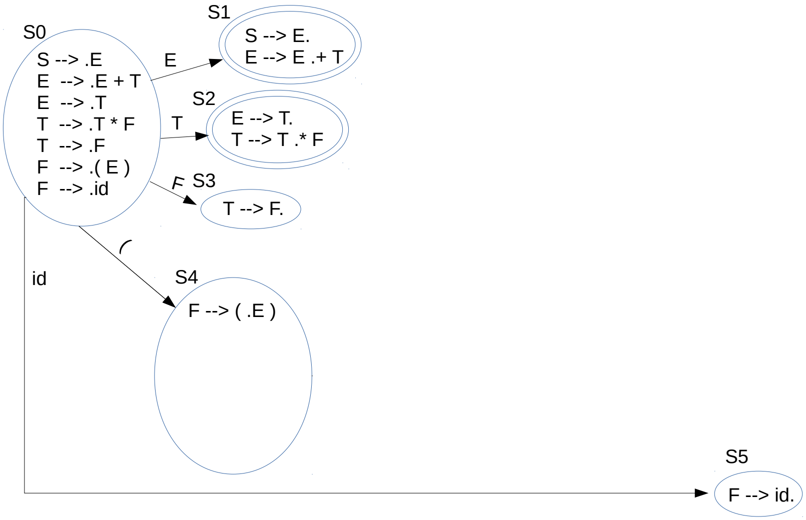
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



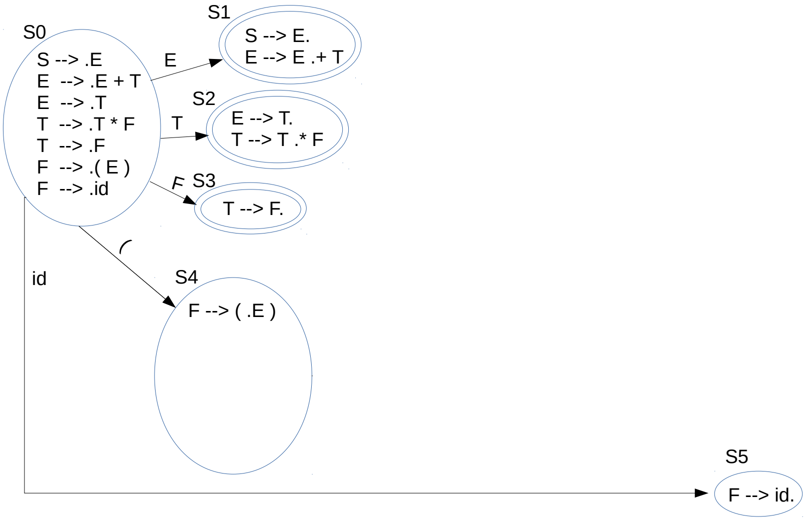
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



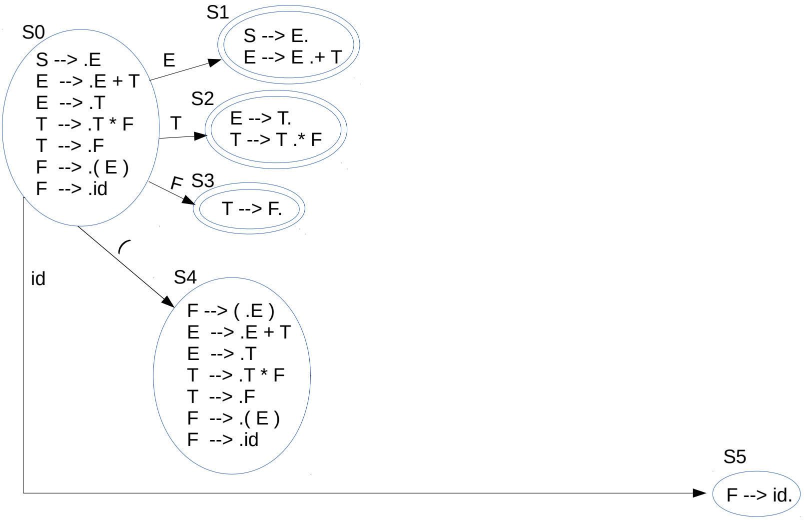
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



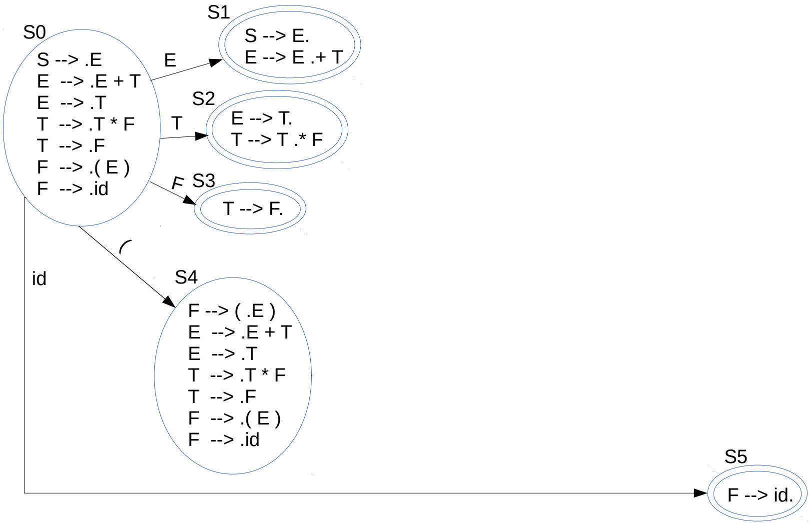
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



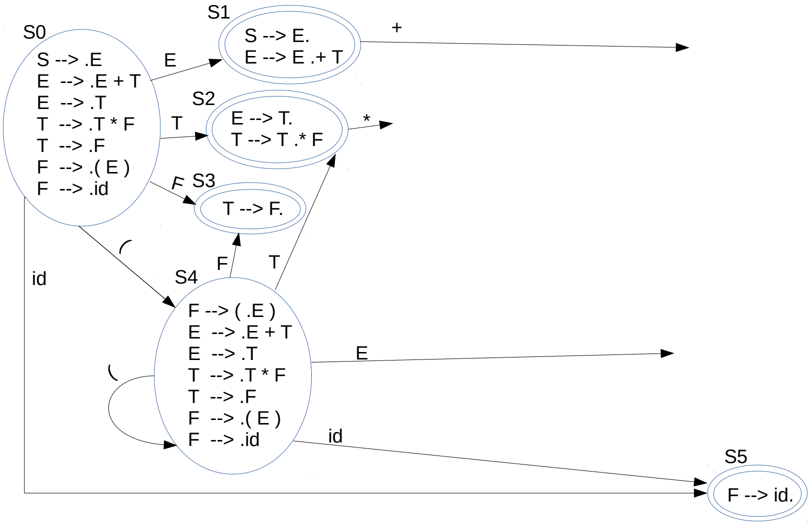
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



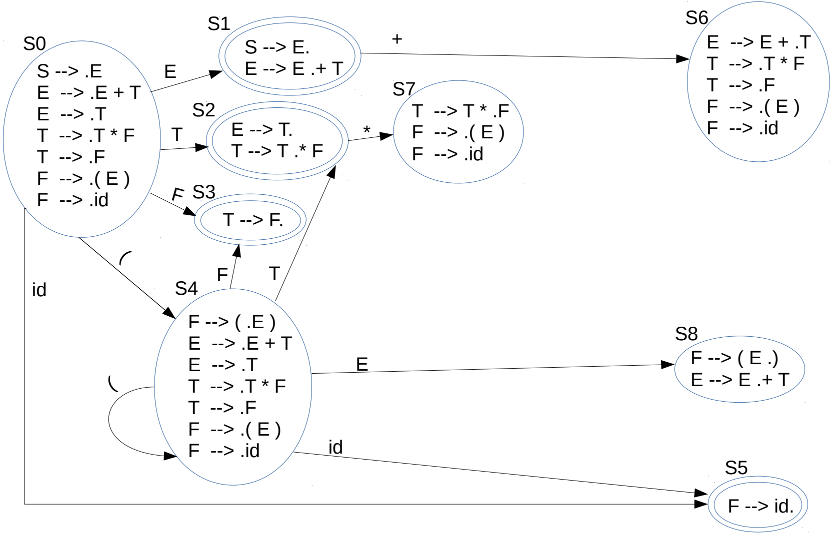
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



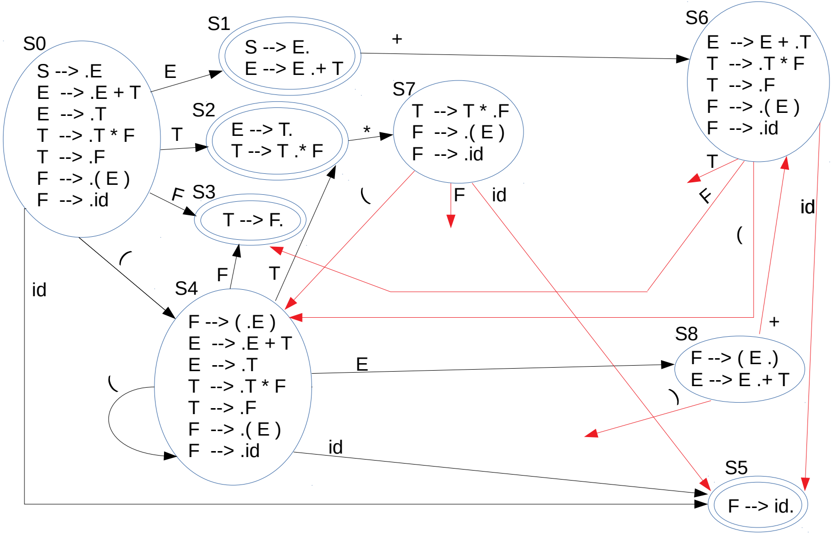
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



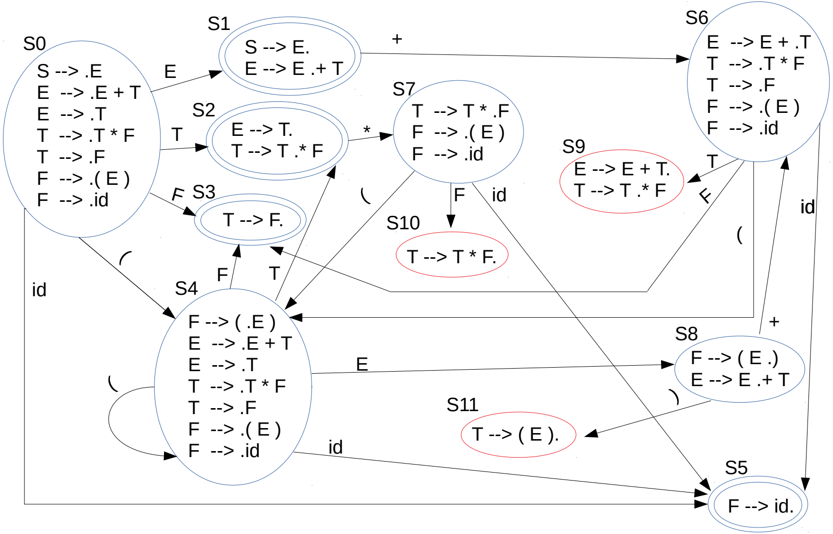
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



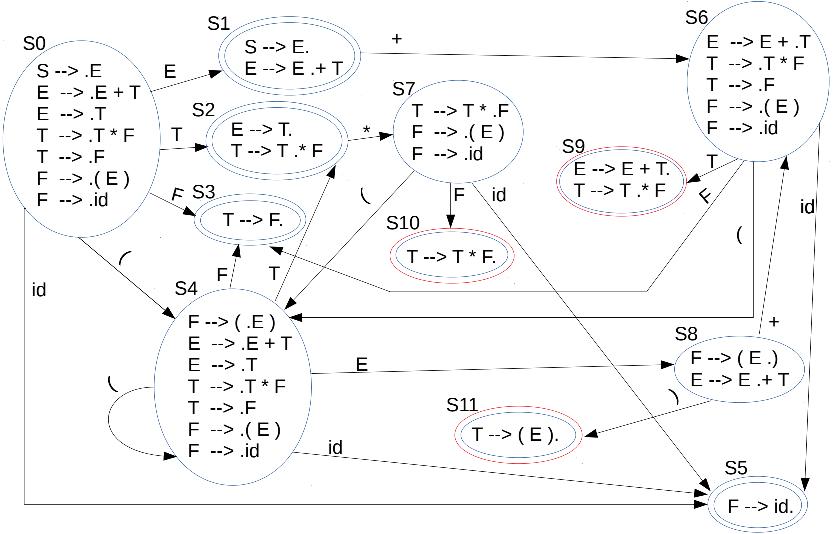
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



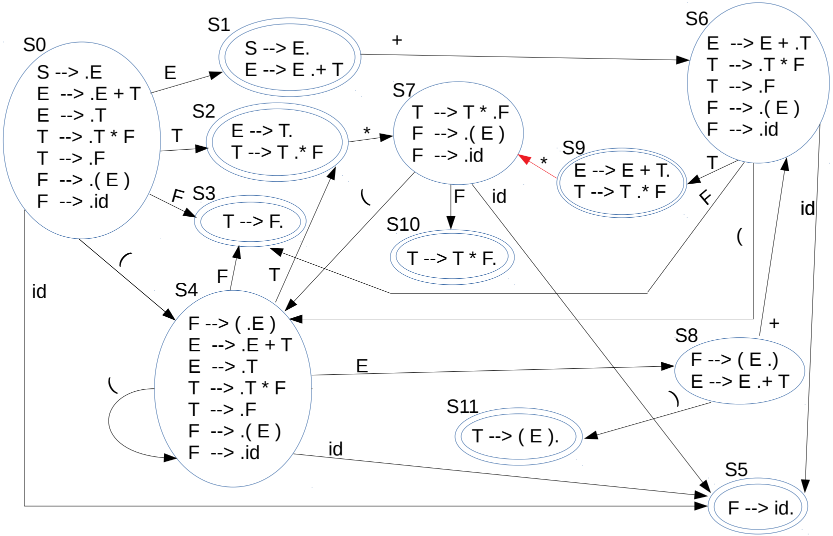
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



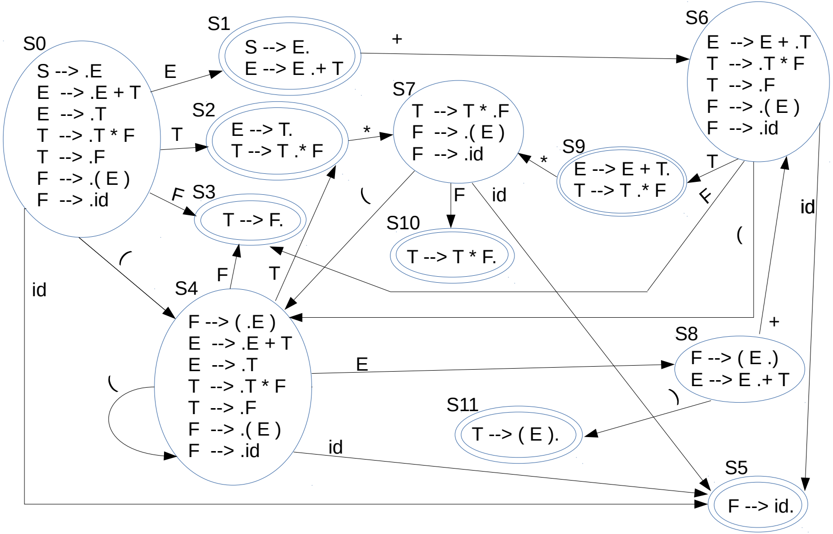
Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



Cálculo directo de $AFD - LR(0)_G$

S --> E , E --> E + T , E --> T , T --> T * F , T --> F , F --> (E) , F --> id



Cálculo directo de $AFD - LR(0)_G$

- Sea $S_0 = clausura(\{[S' \longrightarrow .S]\})$ el primer estado no tratado, donde la clausura de S consiste en añadir los mínimos items tal que si existe $[X \longrightarrow \alpha \bullet Y\beta]$ en S_i entonces todo $[Y \longrightarrow \gamma]$ de P también está en S_i.
- Para todo estado no tratado S_i y para todo N detrás de $\bullet$ en S_i:
  - calculamos la clausura de los sucesores de S_i respecto a N, es decir, $S_j = clausura(\{[X \longrightarrow \alpha N \bullet \beta] \mid [X \longrightarrow \alpha \bullet N\beta] \in S_i\})$
  - Añadimos una transición de S_i a S_j con N.
  - Si S_j es nuevo lo añadimos a los estados no tratados.
  - Una vez considerados todos los sucesores sacamos el estado de los no tratados.

Autómata de pila $LR(0)$

A partir de un $AFD - LR(0)_G$ sin estados inadecuados
$(Q, V_T \cup V_N, \delta, q_0, F)$
creamos el siguiente autómata de pila determinista $LR(0)$

$$(V_T, Q, \Delta, q_0, \{q_f\})$$

- El conjunto de estados (con el que formaremos las pilas) es el mismo de $AFD - LR(0)_G$.
- El estado inicial es la pila con el estado inicial de $AFD - LR(0)_G$
- $q_f = \{[S' \longrightarrow S\bullet]\}$ (en la cima de la pila).
- **Desplazamiento**: Si $[X \longrightarrow \ldots \bullet a \ldots] \in S_i$ y $\delta(S_i, a) = S_j$, entonces
$$(S_i, a, S_i S_j) \in \Delta$$
- **Reducción**: si existe $[X \longrightarrow \alpha\bullet] \in S_i$ con $\mid \alpha \mid = n$ y $\delta(S_k, X) = S_j$ entonces
$$(\underbrace{S_k \ldots S_i}_{n}, \epsilon, S_k S_j) \in \Delta$$

Tabla de análisis para $LR(0)$

La construcción del autómata $LR(0)$ se hace en realidad con una tabla:

- Asignamos el natural i a cada estado S_i de $AFD - LR(0)_G$
- Asignamos el natural q a cada regla de G
- Tabulamos $LR(0)$ de G:

S_i	Acción			Salto		
	a_1	a_2	...	X_1	X_2	...
0		d i	...		k_1	...
$\vdots$						
k	r q	r q	...		k_2	...
$\vdots$						
n	d j		...		k_3	...

- ponemos "d i" en la **acción** (l, a_j) si $\delta(S_l, a_j) = S_i$
- ponemos "r q" en la **acción** (l, a_j) si $[X \longrightarrow \alpha\bullet] \in S_l$ y $X \longrightarrow \alpha$ es la regla q de G.
  En $LR(0)$ en cada fila todas las acciones son el mismo "r q" o ninguna.
- ponemos "k" en el **salto** (l, X_j) si $\delta(S_l, X_j) = S_k$.

Tabla de análisis para $LR(0)$

- Con la tabla T:

S_i	Acción			Salto		
	a_1	a_2	...	X_1	X_2	...
0		d i	...		k_1	...
$\vdots$						
k	r q	r q	...		k_2	...
$\vdots$						
n	d j		...		k_3	...

- Si la **acción** $T[l, a_j]$ es "d i" entonces tenemos un desplazamiento

$$(S_l, a_j, S_l S_i) \in \Delta$$

- Si la **acción** $T[l, a_j]$ es "r i" para todo j y la regla i de G es $X \longrightarrow \alpha$ entonces tenemos una reducción

$$(S_k \underbrace{\ldots S_l}_{n}, \epsilon, S_k S_o) \in \Delta$$

donde $n = \mid \alpha \mid$ y $T[k, X] = o$

es decir, desapilamos n estados y añadimos $S_{T[k,X]}$.

Gramáticas y autómatas LR(k)

Una gramática incontextual G es $LR(k)$, si siempre que se cumple

$$S' \Longrightarrow^*_{de} \alpha X u \Longrightarrow_{de} \alpha\beta u$$
$$S' \Longrightarrow^*_{de} \gamma Y v \Longrightarrow_{de} \alpha\beta w$$
$$k : u = k : w$$

también se cumple que $\alpha = \gamma$, $X = Y$ y $v = w$. es decir, la única posibilidad es tener

$$S' \Longrightarrow^*_{de} \alpha X u \Longrightarrow_{de} \alpha\beta u$$
$$S' \Longrightarrow^*_{de} \alpha X v \Longrightarrow_{de} \alpha\beta v$$
$$k : u = k : v$$

y por tanto, hay una única forma de proceder.

Items para $LR(k)$

- Cuando $AFD - LR(0)$ tiene estados inadecuados, aún es posible construir un analizador determinista si se le permite inspeccionar sin consumirlos hasta k símbolos anticipados de la entrada.
- Añadimos a los items de un **anticipo** (lookahead) que permite discriminar entre dos reducciones, o entre un desplazamiento y una reducción, cuando ambas son posibles.
- Los items de $LR(k)$ son pares $[X \longrightarrow \alpha_1 \bullet \alpha_2, L]$ con $X \longrightarrow \alpha_1\alpha_2 \in P$ y $L \subseteq (V_T \cup \{\vdash\})^+$ de cadenas posiblemente terminadas en $\vdash$ y de longitud menor o igual a k.
  El primer elemento del par se llama **núcleo** y el segundo **anticipo**.
- Decimos que el ítem $[X \longrightarrow \alpha_1 \bullet \alpha_2, L]$ es **válido** para un prefijo viable $\gamma\alpha_1$ si para todo $w \in L$ existe

$$S \Longrightarrow^*_{de} \gamma X u \Longrightarrow_{de} \gamma\alpha_1\alpha_2 u$$

con $w = k : (u \vdash)$.

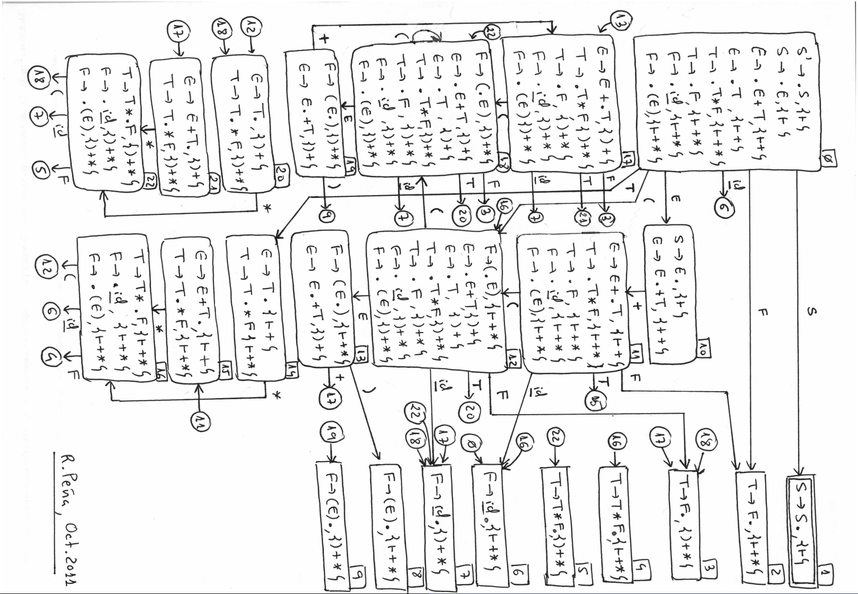
Gramáticas $LR(k)$

- Una gramática G es $LR(k)$ si para todo par de items $[X \longrightarrow \alpha_1 \bullet, L_1]$ y $[Y \longrightarrow \alpha_2 \bullet \alpha_3, L_2]$ válidos para el mismo prefijo viable $\gamma\alpha_1$ tenemos que
$$L_1 \cap prim_k(\alpha_3 L_2) = \emptyset$$
Notad que si α_3 es vacío evitamos conflictos reducción-reducción y si no lo es evitamos conflictos desplazamiento-reducción.
- La construcción del autómata determinista $AFD - LR(k)$ es muy parecida a la de $AFD - LR(0)$, pero usando $prim_k$ en su construcción para mantener el lookahead es sus items.

Cálculo directo de $AFD - LR(k)_G$

- Sea $S_0 = clausura(\{[S' \longrightarrow .S, \{\vdash\}]\})$ el primer estado no tratado, donde la clausura de S consiste en añadir los mínimos items tal que si existe $[X \longrightarrow \alpha \bullet Y\beta, L]$ en S_i entonces $[Y \longrightarrow \gamma, \{prim_k(\beta L)\}]$ también está en S_i para todo $Y \longrightarrow \gamma$.
  Asumimos que todos los items con el mismo núcleo se agrupan uniendo sus anticipos.
- Para todo estado no tratado S_i y para todo N detrás de $\bullet$ en S_i:
  - calculamos la clausura de los sucesores de S_i respecto a N, es decir, $S_j = clausura(\{[X \longrightarrow \alpha N \bullet \beta, L] \mid [X \longrightarrow \alpha \bullet N\beta, L] \in S_i\})$
  - Añadimos una transición de S_i a S_j con N.
  - Si S_j es nuevo lo añadimos a los estados no tratados.
  - Una vez considerados todos los sucesores sacamos el estado de los no tratados.

Ejemplo de $AFD - LR(1)$



De $AFD - LR(k)_G$ a $LR(k)$

Un estado q del $AFD - LR(k)_G$ es **inadecuado** si

- $[X \longrightarrow \alpha\bullet, L_1], [Y \longrightarrow \beta\bullet, L_2] \in q$ y $L_1 \cap L_2 \neq \emptyset$.
  Tenemos un conflicto **reducción-reducción**
- $[X \longrightarrow \alpha\bullet, L_1], [Y \longrightarrow \beta \bullet \gamma, L_2] \in q$ y $L_1 \cap prim_k(\gamma L_2) \neq \emptyset$.
  Tenemos un conflicto **desplazamiento-reducción**

Si todos los estados son **adecuados** entonces la gramática es $LR(k)$ y admite un autómata de pila determinista que la reconoce que se obtienen del $AFD - LR(k)_G$ (con función de transición δ):

- Si en el estado S_i tenemos $[Y \longrightarrow \beta \bullet a\gamma, L]$ y los siguientes k símbolos de la entrada están en $prim_k(a\gamma L)$ entonces tenemos un **desplazamiento**: $(S_i, a, S_iS_j) \in \Delta$ con $\delta(S_i, a) = S_j$
- Si en el estado S_i tenemos $[X \longrightarrow \alpha\bullet, L]$ y los siguientes k símbolos de la entrada están en L entonces tenemos una **reducción**:
  $$(S_k \underbrace{\ldots S_i}_{n}, \epsilon, S_kS_j) \in \Delta \text{ con } \mid \alpha \mid = n \text{ y } \delta(S_k, X) = S_j$$

Tabla de análisis para $LR(1)$

Para el caso $LR(1)$ es como la $LR(0)$ pero podemos tener acciones de desplazamiento y reducción en la misma fila. En el caso de $LR(k)$ la tabla de acciones trabaja con cadenas.

- Tabulamos $LR(1)$ de G:

S_i	Acción			Salto		
	a_1	a_2	$\dots$	X_1	X_2	$\dots$
0		d i	$\dots$		k_1	$\dots$
$\vdots$						
k	d i'	r q	$\dots$		k_2	$\dots$
$\vdots$						
n	d j		$\dots$		k_3	$\dots$

- ponemos "d i" en la **acción** (l, a_j) si $\delta(S_l, a_j) = S_i$
- ponemos "r q" en la **acción** (l, a_j) si $[X \longrightarrow \alpha\bullet, L] \in S_l$, $X \longrightarrow \alpha$ es la regla q de G y $a_j \in L$.
- ponemos "k" en el **salto** (l, X_j) si $\delta(S_l, X_j) = S_k$.

Igual que para $LR(0)$ la tabla determina las transiciones para el autómata $LR(1)$ que reconoce G

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}
P --> .D M {#}

Extendemos con P --> D M

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}
P --> .D M {#}
P --> €. {#}

Extendemos con P --> €

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}
P --> .D M {#}
P --> €. {#}
D --> .D a {prim(M#)}

Extendemos con D --> D a

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}
P --> .D M {#}
P --> €. {#}
D --> .D a {a}

Extendemos con D --> D a

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}
P --> .D M {#}
P --> €. {#}
D --> .D a {a}
D --> €. {a}

Extendemos con D --> €

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}
P --> .D M {#}
P --> €. {#}
D --> .D a {a}
D --> €. {a}

Extendemos con D --> D a pero en D --> .D a

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

P' --> .P {#}
P --> .D M {#}
P --> €. {#}
D --> .D a {a}
D --> €. {a}

Extendemos con D --> € pero en D --> .D a

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P

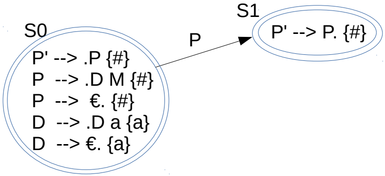
S0

P' --> .P {#}
P --> .D M {#}
P --> €. {#}
D --> .D a {a}
D --> €. {a}

Hemos terminado la clausura del estado que es final

Ejemplo de cálculo de $AFD - LR(1)$

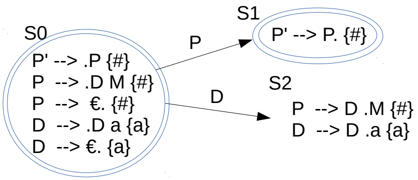
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Añadimos la transición desde S0 con P y nos da
un estado final que está cerrado

Ejemplo de cálculo de $AFD - LR(1)$

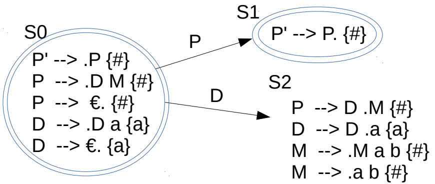
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Añadimos la transición desde S0 con D y nos da un estado que hay que cerrar

Ejemplo de cálculo de $AFD - LR(1)$

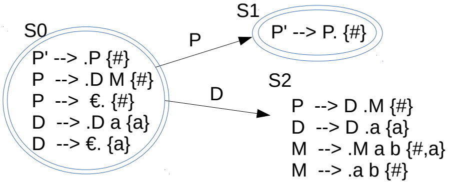
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Extendemos con M --> M a b y con M --> a b

Ejemplo de cálculo de $AFD - LR(1)$

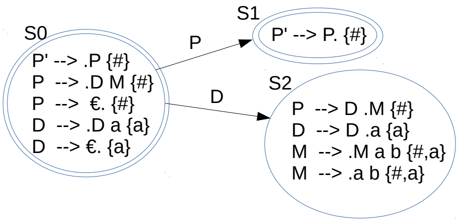
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Extendemos M --> M a b pero en M --> .M a b

Ejemplo de cálculo de $AFD - LR(1)$

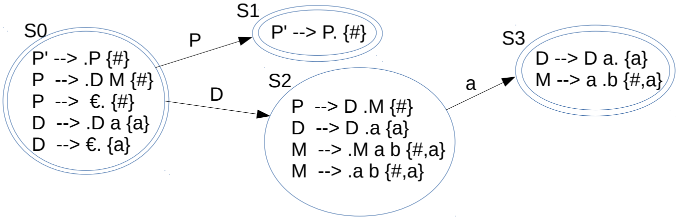
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Hemos terminado la clausura del estado

Ejemplo de cálculo de $AFD - LR(1)$

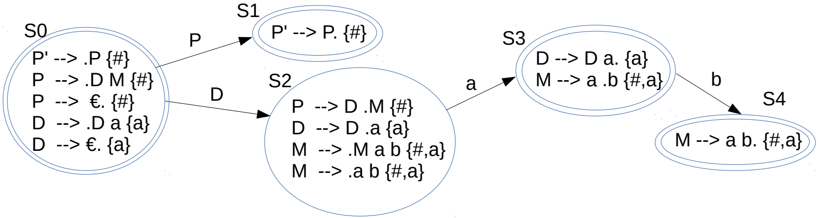
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Añadimos la transición desde S2 con a y nos da
un estado final que está cerrado

Ejemplo de cálculo de $AFD - LR(1)$

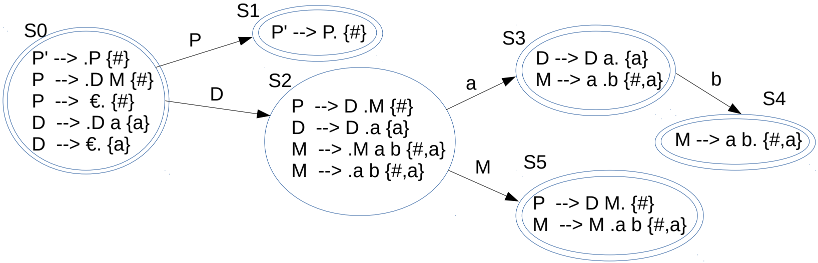
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Añadimos la transición desde S3 con b y nos da
un estado final que está cerrado

Ejemplo de cálculo de $AFD - LR(1)$

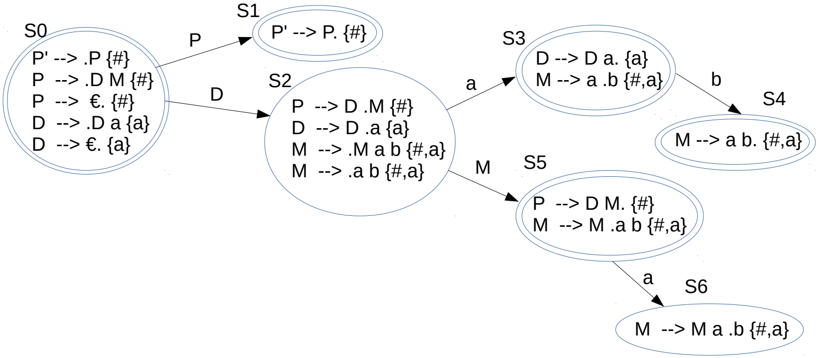
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Añadimos la transición desde S2 con M y nos da
un estado final que está cerrado

Ejemplo de cálculo de $AFD - LR(1)$

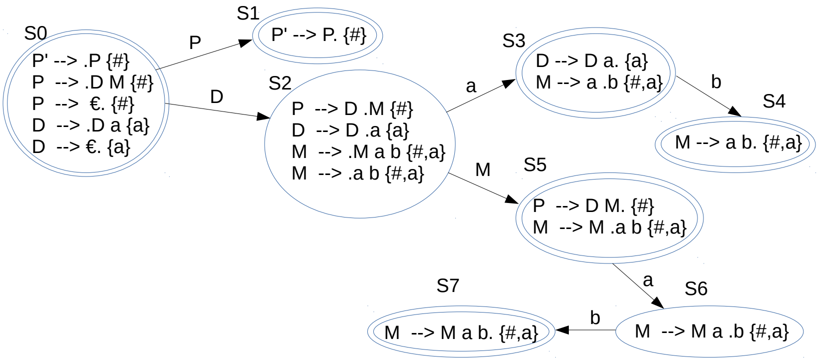
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Añadimos la transición desde S5 con a y nos da
un estado que está cerrado

Ejemplo de cálculo de $AFD - LR(1)$

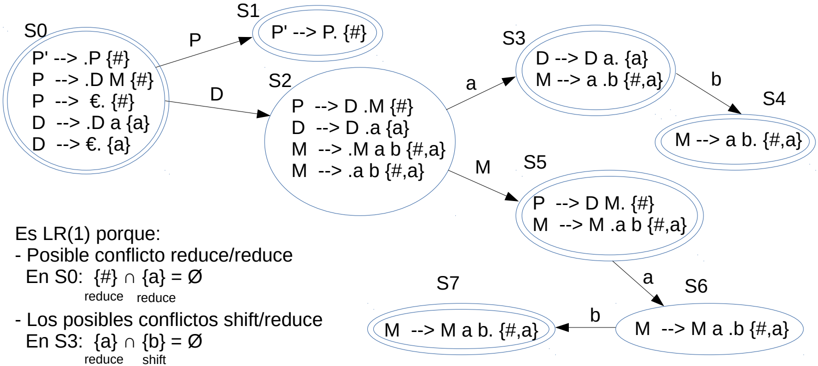
P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Añadimos la transición desde S6 con b y nos da un estado final cerrado y hemos terminado

Ejemplo de cálculo de *AFD* – *LR*(1)

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Es LR(1) porque:

- Posible conflicto reduce/reduce

  En S0: $\underbrace{\{\#\}}_{\text{reduce}} \cap \underbrace{\{a\}}_{\text{reduce}} = \emptyset$

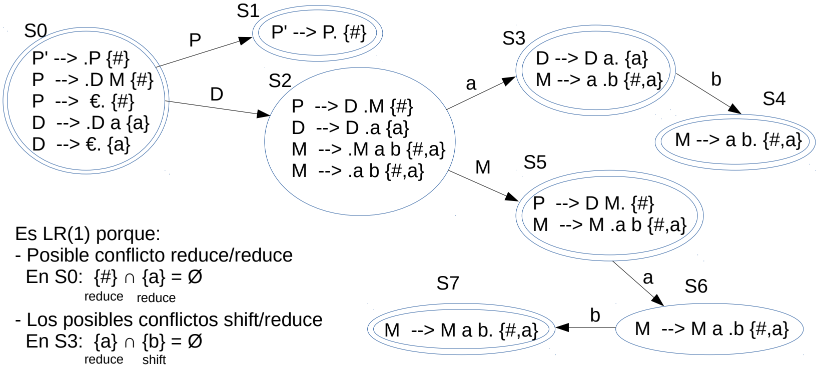
- Los posibles conflictos shift/reduce

  En S3: $\underbrace{\{a\}}_{\text{reduce}} \cap \underbrace{\{b\}}_{\text{shift}} = \emptyset$

  En S5: $\underbrace{\{\#\}}_{\text{reduce}} \cap \underbrace{\{a\}}_{\text{shift}} = \emptyset$

Ejemplo de cálculo de $AFD - LR(1)$

P --> D M , P --> € , D --> D a , D --> € , M --> M a b , M --> a b , P' --> P



Es LR(1) porque:

- Posible conflicto reduce/reduce

  En S0: $\underbrace{\{\#\}}_{\text{reduce}} \cap \underbrace{\{a\}}_{\text{reduce}} = \emptyset$

- Los posibles conflictos shift/reduce

  En S3: $\underbrace{\{a\}}_{\text{reduce}} \cap \underbrace{\{b\}}_{\text{shift}} = \emptyset$

  En S5: $\underbrace{\{\#\}}_{\text{reduce}} \cap \underbrace{\{a\}}_{\text{shift}} = \emptyset$

Construid la tabla de acciones que determina el autómata de pila para el parser LR(1)

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

Como reconoce, por ejemplo, $aaabab\#$?

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$\boxed{\text{a}}\, aabab\#$
S_0

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon \qquad |\epsilon| = 0$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$\boxed{\text{a}}\, aabab\#$

S_0

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon \qquad |\epsilon| = 0$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$\boxed{\text{a}}\, aabab\#$
S_0

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$\boxed{\text{a}}\,aabab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$\boxed{\text{a}}\, aabab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$a\boxed{\text{a}}abab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a \quad |D\ a| = 2$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$a\boxed{a}abab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a \qquad |D\ a| = 2$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$a\boxed{\text{a}}abab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$a\boxed{a}abab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$a\boxed{a}abab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aa\boxed{\text{a}}bab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a \qquad |D\ a| = 2$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aa\boxed{\mathrm{a}}bab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a \qquad |D\ a| = 2$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aa\boxed{a}bab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aa\boxed{\text{a}}bab\#$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aa\boxed{\text{a}}bab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaa\boxed{\text{b}}ab\#$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aaa\boxed{\text{b}}ab\#$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2{\color{red}S_3}$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaab\boxed{\text{a}}b\#$

$$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b \quad |a\ b| = 2$

Estado inicial: S_0

Estado final: S_1

$aaab\boxed{a}b\#$

$$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b \quad |a\ b| = 2$

Estado inicial: S_0

Estado final: S_1

$aaab\boxed{a}b\#$

$$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{b} S_0 S_2 S_3 S_4$$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaab\boxed{\text{a}}b\#$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4 \xrightarrow{\epsilon} S_0S_2S_5$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aaab\boxed{\text{a}}b\#$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4 \xrightarrow{\epsilon} S_0S_2S_5$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aaaba\boxed{\text{b}}\#$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$
$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaaba\boxed{\text{b}}\#$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$
$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aaabab\boxed{\#}$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$
$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b \qquad |M\ a\ b| = 3$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaabab\boxed{\#}$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{b} S_0 S_2 S_3 S_4$
$\xrightarrow{\epsilon} S_0 S_2 S_5 \xrightarrow{a} S_0 S_2 S_5 S_6 \xrightarrow{b} S_0 S_2 S_5 S_6 S_7$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b \qquad |M\ a\ b| = 3$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaabab\boxed{\#}$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4 \xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aaabab\boxed{\#}$

$$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4 \xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7 \xrightarrow{\epsilon} S_0S_2S_5$$

29

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$

1) $P \longrightarrow D\ M \qquad |D\ M| = 2$

2) $P \longrightarrow \epsilon$

3) $D \longrightarrow D\ a$

4) $D \longrightarrow \epsilon$

5) $M \longrightarrow M\ a\ b$

6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaabab\boxed{\#}$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$
$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7 \xrightarrow{\epsilon} S_0S_2S_5$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M \qquad |D\ M| = 2$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaabab\boxed{\#}$

$S_0 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{\epsilon} S_0 S_2 \xrightarrow{a} S_0 S_2 S_3 \xrightarrow{b} S_0 S_2 S_3 S_4$
$\xrightarrow{\epsilon} S_0 S_2 S_5 \xrightarrow{a} S_0 S_2 S_5 S_6 \xrightarrow{b} S_0 S_2 S_5 S_6 S_7 \xrightarrow{\epsilon} S_0 S_2 S_5$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$

1) $P \longrightarrow D\ M$

2) $P \longrightarrow \epsilon$

3) $D \longrightarrow D\ a$

4) $D \longrightarrow \epsilon$

5) $M \longrightarrow M\ a\ b$

6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaabab\boxed{\#}$

$$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$$
$$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7 \xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{\epsilon} S_0S_1$$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P \qquad |P| = 1$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaabab\boxed{\#}$

$$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$$
$$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7 \xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{\epsilon} S_0S_1$$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P \quad |P| = 1$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0

Estado final: S_1

$aaabab\boxed{\#}$

$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$
$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7 \xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{\epsilon} S_0S_1$

Tabla de acciones

S_i	Acción			Salto		
	a	b	#	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aaabab\boxed{\#}$

$$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$$
$$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7 \xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{\epsilon} S_0S_1 \xrightarrow{\epsilon} S_0$$

Tabla de acciones

S_i	Acción			Salto		
	a	b	$\#$	P	D	M
0	r4		r2	1	2	
1			r0			
2	d3					5
3	r3	d4				
4	r6		r6			
5	d6		r1			
6		d7				
7	r5		r5			

0) $P' \longrightarrow P$
1) $P \longrightarrow D\ M$
2) $P \longrightarrow \epsilon$
3) $D \longrightarrow D\ a$
4) $D \longrightarrow \epsilon$
5) $M \longrightarrow M\ a\ b$
6) $M \longrightarrow a\ b$

Estado inicial: S_0
Estado final: S_1

$aaabab\boxed{\#}$

$$S_0 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{\epsilon} S_0S_2 \xrightarrow{a} S_0S_2S_3 \xrightarrow{b} S_0S_2S_3S_4$$
$$\xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{a} S_0S_2S_5S_6 \xrightarrow{b} S_0S_2S_5S_6S_7 \xrightarrow{\epsilon} S_0S_2S_5 \xrightarrow{\epsilon} S_0S_1 \xrightarrow{\epsilon} S_0 \quad \text{acepta}$$

Métodos simplificados $SLR(1)$ y $LALR(1)$

Estos métodos pretender evitar la potencial explosión de estados del autómata $AFD - LR(1)$.

Por esa razón se basan en $AFD - LR(0)$.

Para el método $SLR(1)$ (de *Simple* LR(1)), cada ítem $[X \longrightarrow \alpha\bullet]$ en $AFD - LR(0)$ se reemplaza por $[X \longrightarrow \alpha\bullet, sig(X)]$.

Para la gramática

$S \longrightarrow E$
$E \longrightarrow E + T \mid T$
$T \longrightarrow T * F \mid F$
$F \longrightarrow "(" \; E \; ")" \mid \mathbf{id}$

S_1	S_2	S_9
$[S \longrightarrow E\bullet, \{\vdash\}]$	$[E \longrightarrow T\bullet, \{\vdash,), +\}]$	$[S \longrightarrow E + T\bullet, \{\vdash,), +\}]$
$[E \longrightarrow E \bullet + T]$	$[T \longrightarrow T \bullet * F]$	$[T \longrightarrow T \bullet * F]$

Los tres estados son adecuados en el sentido $LR(1)$.

Si todos los estados resultantes son adecuados decimos que G es $SLR(1)$.

Métodos simplificados $SLR(1)$ y $LALR(1)$

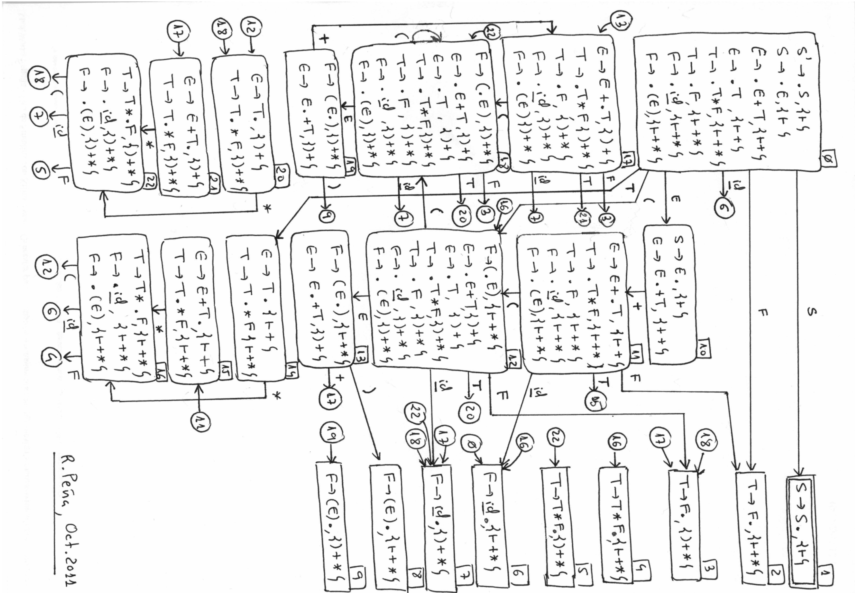
En el método $LALR(1)$ (de *Look Ahead* $LR(1)$) cada ítem $[X \longrightarrow \alpha\bullet]$ en $AFD - LR(0)$ se reemplaza por $[X \longrightarrow \alpha\bullet, \bigcup_i L_i]$, donde los L_i son anticipos de los items con núcleo $X \longrightarrow \alpha\bullet$ de los estados homólogos en $AFD - LR(1)$.

Donde dos estados son homólogos si son iguales al quitar los anticipos (es decir, como estados $AFD - LR(0)$).

- se puede obtener el autómata $LALR(1)$ fusionando los estados homólogos de $AFD - LR(1)$
- se puede obtener directamente, calculado un conjunto de ecuaciones sobre anticipos cuya solución nos da los anticipos para $LALR(1)$ (ver notas del curso).

Para el ejemplo anterior las parejas de estados $(2,3)$, $(4,5)$ $(6,7)$ $(8,9)$, $(11,17)$, $(12,18)$, $(13,19)$, $(14,20)$, $(15,21)$ y $(16,22)$ son homólogos.

Métodos simplificados $SLR(1)$ y $LALR(1)$



Métodos simplificados $SLR(1)$ y $LALR(1)$

Los potenciales estados conflictivos son el 10 y la fusión de los estados 14 y 20 y los estados 15 y 21, con lo que tenemos:

S_{10}	$S_{14}S_{20}$	$S_{15}S_{21}$
$[S \longrightarrow E\bullet, \{\vdash\}]$	$[E \longrightarrow T\bullet, \{\vdash,), +\}]$	$[S \longrightarrow E + T\bullet, \{\vdash,), +\}]$
$[E \longrightarrow E \bullet + T]$	$[T \longrightarrow T \bullet * F]$	$[T \longrightarrow T \bullet * F]$

Los tres estados son adecuados.

Si todos los estados resultantes son adecuados decimos que G es $LALR(1)$.

En general se cumple:

$$LR(0) \subset SLR(1) \subset LALR(1) \subset LR(1)$$

Contenidos

Analizadores $LL(k)$

Un analizador descendente $LL(k)$ es un autómata de items al que se ha eliminado el no-determinismo de la regla de expansión.

Analizadores $LL(k)$

Un analizador descendente $LL(k)$ es un autómata de items al que se ha eliminado el no-determinismo de la regla de expansión.

Stack contents	Remaining input
$[S \to .E]$	$\mathbf{id} + \mathbf{id} * \mathbf{id}$
$[S \to .E][E \to .E+T]$	$\mathbf{id} + \mathbf{id} * \mathbf{id}$
$[S \to .E][E \to .E+T][E \to .T]$	$\mathbf{id} + \mathbf{id} * \mathbf{id}$
$[S \to .E][E \to .E+T][E \to .T][T \to .F]$	$\mathbf{id} + \mathbf{id} * \mathbf{id}$
$[S \to .E][E \to .E+T][E \to .T][T \to .F][F \to .\mathbf{id}]$	$\mathbf{id} + \mathbf{id} * \mathbf{id}$
$[S \to .E][E \to .E+T][E \to .T][T \to .F][F \to \mathbf{id}.]$	$+\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to .E+T][E \to .T][T \to F.]$	$+\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to .E+T][E \to T.]$	$+\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to E.+T]$	$+\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to E+.T]$	$\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to E+.T][T \to .T*F]$	$\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to E+.T][T \to .T*F][T \to .F]$	$\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to E+.T][T \to .T*F][T \to .F][F \to .\mathbf{id}]$	$\mathbf{id} * \mathbf{id}$
$[S \to .E][E \to E+.T][T \to .T*F][T \to .F][F \to \mathbf{id}.]$	$*\mathbf{id}$
$[S \to .E][E \to E+.T][T \to .T*F][T \to F.]$	$*\mathbf{id}$
$[S \to .E][E \to E+.T][T \to T.*F]$	$*\mathbf{id}$
$[S \to .E][E \to E+.T][T \to T*.F]$	$\mathbf{id}$
$[S \to .E][E \to E+.T][T \to T*.F][F \to .\mathbf{id}]$	$\mathbf{id}$
$[S \to .E][E \to E+.T][T \to T*.F][F \to \mathbf{id}.]$	
$[S \to .E][E \to E+.T][T \to T*F.]$	
$[S \to .E][E \to E+T.]$	
$[S \to E.]$	

Analizadores $LL(k)$

Un analizador descendente $LL(k)$ es un autómata de items al que se ha eliminado el no-determinismo de la regla de expansión. Para ello, se le permite inspeccionar anticipadamente (sin consumirlos) los primeros k símbolos de (lo que queda de) la entrada.

Una gramática es $LL(k)$ si para todo par de derivaciones por la izquierda

$$S \Longrightarrow^*_{iz} uY\alpha \Longrightarrow^*_{iz} u\beta\alpha \Longrightarrow^*_{iz} ux$$
$$S \Longrightarrow^*_{iz} uY\alpha \Longrightarrow^*_{iz} u\gamma\alpha \Longrightarrow^*_{iz} uy$$

si se cumple que $k : x = k : y$ entonces también se cumple que $\beta = \gamma$.

Es decir, la alternativa elegida para expandir un no-terminal Y está determinada de forma única por u y los primeros k símbolos de la cadena remanente.

Analizadores $LL(1)$

Cuando $k = 1$, G es $LL(1)$ si y solo si para todo conjunto de reglas $Y \longrightarrow \alpha_1 \mid \ldots \mid \alpha_n$ tenemos

- $prim(\alpha_i) \cap prim(\alpha_j) = \emptyset$ para todo $1 \leq i < j \leq n$,
- si existe i tal que $\epsilon \in prim(\alpha_i)$ entonces $prim(\alpha_j) \cap sig(Y) = \emptyset$ para todo $j \neq i$

La siguiente gramática de expresiones es $LL(1)$:

$$
\begin{array}{lll}
S \longrightarrow E & T \longrightarrow FT' \\
E \longrightarrow TE' & T' \longrightarrow *T \mid \epsilon \\
E' \longrightarrow +E \mid \epsilon & F \longrightarrow "(" \; E \; ")" \mid \textbf{id}
\end{array}
$$

$$
\begin{array}{l}
prim(+E) \cap prim(\epsilon) = \emptyset \\
prim(*T) \cap prim(\epsilon) = \emptyset \\
prim(\underline{(}E\underline{)}) \cap prim(id) = \emptyset \\
prim(+E) \cap sig(E') = \emptyset \\
prim(*T) \cap sig(T') = \emptyset
\end{array}
$$

	S	E	E'	T	T'	F
prim(X)	$\{\underline{(}, id\}$	$\{\underline{(}, id\}$	$\{+, \epsilon\}$	$\{\underline{(}, id\}$	$\{*, \epsilon\}$	$\{\underline{(}, id\}$
sig(X)	$\{\vdash\}$	$\{\vdash \underline{)}\}$	$\{\vdash, \underline{)}\}$	$\{+, \vdash, \underline{)}\}$	$\{+, \vdash, \underline{)}\}$	$\{*, +, \vdash, \underline{)}\}$

Tabla predictora LL(1)

Un analizador $LL(1)$ para G es el autómata (de pila) de items K_G determinizado con una tabla predictora:

$$m : V_N \times V_T \to P \cup \{error\}$$

que determina la regla a aplicar cuando el ítem en la cima es $[X \longrightarrow \beta \bullet Y\gamma]$ y en la entrada tenemos a:

- Si $m[Y, a] = (Y \longrightarrow \alpha)$ entonces hay que expandir Y con $Y \longrightarrow \alpha$.
- Si $m[Y, a] = error$ la cadena no pertenece al lenguaje.

El algoritmo para rellenar la tabla predictora para una gramática $LL(1)$:

1. $m[X, a] = (X \longrightarrow \alpha)$ si $(X \longrightarrow \alpha) \in P$ y $a \neq \epsilon \in prim(\alpha)$.
2. $m[X, b] = (X \longrightarrow \alpha)$ si $(X \longrightarrow \alpha) \in P$, $\epsilon \in prim(\alpha)$ y $b \in sig(X)$.
3. El resto de entradas $m[Y, c] = error$.

Tabla predictora LL(1)

Usando

	S	E	E'	T	T'	F
prim(X)	$\{\underline{(}, id\}$	$\{\underline{(}, id\}$	$\{+, \epsilon\}$	$\{\underline{(}, id\}$	$\{*, \epsilon\}$	$\{\underline{(}, id\}$
sig(X)	$\{\vdash\}$	$\{\vdash \underline{)}\}$	$\{\vdash, \underline{)}\}$	$\{+, \vdash, \underline{)}\}$	$\{+, \vdash, \underline{)}\}$	$\{*, +, \vdash, \underline{)}\}$

tenemos

	(	)	$+$	$*$	id	$\vdash$
E	$E \longrightarrow TE'$	error	error	error	$E \longrightarrow TE'$	error
E'	error	$E' \longrightarrow \epsilon$	$E' \longrightarrow +E$	error	error	$E' \longrightarrow \epsilon$
T	$T \longrightarrow FT'$	error	error	error	$T \longrightarrow FT'$	error
T'	error	$T' \longrightarrow \epsilon$	$T' \longrightarrow \epsilon$	$T' \longrightarrow *T$	error	$T' \longrightarrow \epsilon$
F	$F \longrightarrow \underline{(}\ E\ \underline{)}$	error	error	error	$F \longrightarrow \mathbf{id}$	error
S	$S \longrightarrow E$	error	error	error	$S \longrightarrow E$	error

Tabla predictora LL(1)

Usando

	S	E	E'	T	T'	F
prim(X)	$\{\underline{(}, id\}$	$\{\underline{(}, id\}$	$\{+, \epsilon\}$	$\{\underline{(}, id\}$	$\{*, \epsilon\}$	$\{\underline{(}, id\}$
sig(X)	$\{\#\}$	$\{\#\underline{)}\}$	$\{\#, \underline{)}\}$	$\{+, \#, \underline{)}\}$	$\{+, \#, \underline{)}\}$	$\{*, +, \#, \underline{)}\}$

tenemos

	(	)	$+$	$*$	id	$\#$
E	$E \longrightarrow TE'$	error	error	error	$E \longrightarrow TE'$	error
E'	error	$E' \longrightarrow \epsilon$	$E' \longrightarrow +E$	error	error	$E' \longrightarrow \epsilon$
T	$T \longrightarrow FT'$	error	error	error	$T \longrightarrow FT'$	error
T'	error	$T' \longrightarrow \epsilon$	$T' \longrightarrow \epsilon$	$T' \longrightarrow *T$	error	$T' \longrightarrow \epsilon$
F	$F \longrightarrow \underline{(}\ E\ \underline{)}$	error	error	error	$F \longrightarrow$ **id**	error
S	$S \longrightarrow E$	error	error	error	$S \longrightarrow E$	error

También podemos usar el símbolo $\#$ en lugar de $\vdash$ para el final de entrada.

Tabla predictora LL(1)

Usando

	S	E	E'	T	T'	F
prim(X)	$\{\underline{(}, id\}$	$\{\underline{(}, id\}$	$\{+, \epsilon\}$	$\{\underline{(}, id\}$	$\{*, \epsilon\}$	$\{\underline{(}, id\}$
sig(X)	$\{\#\}$	$\{\#\underline{)}\}$	$\{\#, \underline{)}\}$	$\{+, \#, \underline{)}\}$	$\{+, \#, \underline{)}\}$	$\{*, +, \#, \underline{)}\}$

tenemos

	(	)	$+$	$*$	id	$\#$
E	$E \longrightarrow TE'$	error	error	error	$E \longrightarrow TE'$	error
E'	error	$E' \longrightarrow \epsilon$	$E' \longrightarrow +E$	error	error	$E' \longrightarrow \epsilon$
T	$T \longrightarrow FT'$	error	error	error	$T \longrightarrow FT'$	error
T'	error	$T' \longrightarrow \epsilon$	$T' \longrightarrow \epsilon$	$T' \longrightarrow *T$	error	$T' \longrightarrow \epsilon$
F	$F \longrightarrow \underline{(}\ E\ \underline{)}$	error	error	error	$F \longrightarrow$ **id**	error
S	$S \longrightarrow E$	error	error	error	$S \longrightarrow E$	error

También podemos usar el símbolo $\#$ en lugar de $\vdash$ para el final de entrada.

Con esta tabla el reconocimiento de $id * id\#$ aplica las siguientes reglas: $(S \longrightarrow E)(E \longrightarrow TE')(T \longrightarrow FT')(F \longrightarrow id)(T' \longrightarrow *T)(T \longrightarrow FT')(F \longrightarrow id)(T' \longrightarrow \epsilon)(E' \longrightarrow \epsilon)$

Tabla predictora LL(1)

	(	)	$+$	$*$	id	$\#$
E	$E \longrightarrow TE'$	error	error	error	$E \longrightarrow TE'$	error
E'	error	$E' \longrightarrow \epsilon$	$E' \longrightarrow +E$	error	error	$E' \longrightarrow \epsilon$
T	$T \longrightarrow FT'$	error	error	error	$T \longrightarrow FT'$	error
T'	error	$T' \longrightarrow \epsilon$	$T' \longrightarrow \epsilon$	$T' \longrightarrow *T$	error	$T' \longrightarrow \epsilon$
F	$F \longrightarrow (E)$	error	error	error	$F \longrightarrow \mathbf{id}$	error
S	$S \longrightarrow E$	error	error	error	$S \longrightarrow E$	error

Stack contents	Input
$\#[S \to .E]$	$\mathbf{id} * \mathbf{id}\#$
$\#[S \to .E][E \to .TE']$	$\mathbf{id} * \mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to .FT']$	$\mathbf{id} * \mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to .FT'][F \to .\mathbf{id}]$	$\mathbf{id} * \mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to .FT'][F \to \mathbf{id}.]$	$*\mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to F.T']$	$*\mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to .*T]$	$*\mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *.T]$	$\mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *.T][T \to .FT']$	$\mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *.T][T \to .FT'][F \to .\mathbf{id}]$	$\mathbf{id}\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *.T][T \to .FT'][F \to \mathbf{id}.]$	$\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *.T][T \to F.T']$	$\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *.T][T \to F.T'][T' \to \varepsilon.]$	$\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *.T][T \to FT'.]$	$\#$
$\#[S \to .E][E \to .TE'][T \to F.T'][T' \to *T.]$	$\#$
$\#[S \to .E][E \to .TE'][T \to FT'.]$	$\#$
$\#[S \to .E][E \to T.E']$	$\#$
$\#[S \to .E][E \to T.E'][E' \to \varepsilon.]$	$\#$
$\#[S \to .E][E \to TE'.]$	$\#$
$\#[S \to E.]$	$\#$
$\#$	$\#$

Transformación de gramáticas a $LL(1)$

Una gramática recursiva por la izquierda no es $LL(k)$:

- G es recursiva por la izquierda si tiene una regla $A \longrightarrow A\beta$
- Como G es reducida, tendrá otra regla $A \longrightarrow \gamma$ (si no, A sería improductivo). Por tanto, dado que A es alcanzable tiene que existir una derivación por la izquierda de la forma:
  $S \Longrightarrow^*_{iz} uA\alpha \Longrightarrow^*_{iz} uA\beta^n\alpha$
  Para poder elegir deterministamente entre aplicar $A \longrightarrow A\beta$ o $A \longrightarrow \gamma$ necesitamos que
  $prim_k(A\beta^{n+1}\alpha) \cap prim_k(\gamma\beta^n\alpha) = \emptyset$,
  lo que implica por definición de $prim_k$ que
  $prim_k(\gamma\beta^{n+1}\alpha) \cap prim_k(\gamma\beta^n\alpha) = \emptyset$,
  lo que, tomando n suficientemente grande, no se puede cumplir.

Eliminar recursión por la izquierda

Siempre se puede transformar las gramáticas recursivas por la izquierda en gramáticas recursivas por la derecha.

La transformación se basa en sustituir las reglas
$A \longrightarrow A\alpha_1 \mid \ldots \mid A\alpha_n \mid \beta_1 \mid \ldots \mid \beta_m$
por las reglas:

$$\begin{array}{lcl} A & \longrightarrow & \beta_1 A' \mid \ldots \mid \beta_m A' \\ A' & \longrightarrow & \alpha_1 A' \mid \ldots \alpha_n A' \mid \epsilon \end{array}$$

Puede extenderse fácilmente a recursividad **indirecta** por la izquierda, es decir con derivaciones $A \Longrightarrow_G^+ A\alpha$.

Fijamos inicialmente un orden cualquiera entre los no terminales.

En cada paso eliminamos las reglas $A_i \longrightarrow A_j\alpha$ con A_j menor o igual que A_j en el orden.

Eliminar recursión por la izquierda

Consideramos la gramática recursiva por la izquierda:

$$\left\{ \begin{array}{l} E \longrightarrow E + T \ \mid \ T \\ T \longrightarrow T * F \mid F \\ F \longrightarrow \underline{(} \ E \ \underline{)} \mid \mathbf{id} \end{array} \right.$$

al eliminar la recursividad por la izquierda obtenemos

Eliminar recursión por la izquierda

Consideramos la gramática recursiva por la izquierda:

$$\left\{\begin{array}{l} E \longrightarrow E + T \ \mid \ T \\ T \longrightarrow T * F \ \mid \ F \\ F \longrightarrow \underline{(}\ E\ \underline{)} \ \mid \ \mathbf{id} \end{array}\right.$$

al eliminar la recursividad por la izquierda obtenemos

$$\left\{\begin{array}{ll} E \longrightarrow TE' & \qquad E' \longrightarrow +TE' \ \mid \ \epsilon \\ & \end{array}\right.$$

Eliminar recursión por la izquierda

Consideramos la gramática recursiva por la izquierda:

$$\left\{ \begin{array}{l} E \longrightarrow E + T \mid T \\ T \longrightarrow T * F \mid F \\ F \longrightarrow \underline{(}\ E\ \underline{)} \mid \mathbf{id} \end{array} \right.$$

al eliminar la recursividad por la izquierda obtenemos

$$\left\{ \begin{array}{ll} E \longrightarrow TE' & E' \longrightarrow +TE' \mid \epsilon \\ T \longrightarrow FT' & T' \longrightarrow *FT' \mid \epsilon \\ & \end{array} \right.$$

Eliminar recursión por la izquierda

Consideramos la gramática recursiva por la izquierda:

$$\left\{\begin{array}{l} E \longrightarrow E + T \ \mid \ T \\ T \longrightarrow T * F \ \mid \ F \\ F \longrightarrow \underline{(}\ E\ \underline{)} \ \mid \ \mathbf{id} \end{array}\right.$$

al eliminar la recursividad por la izquierda obtenemos

$$\left\{\begin{array}{ll} E \longrightarrow TE' & E' \longrightarrow +TE' \ \mid \ \epsilon \\ T \longrightarrow FT' & T' \longrightarrow *FT' \ \mid \ \epsilon \\ F \longrightarrow \underline{(}\ E\ \underline{)} \ \mid \ \mathbf{id} & \end{array}\right.$$

Factorización por la izquierda

Una gramática no está factorizada por izquierda si tiene reglas de la forma: $A \longrightarrow \alpha\beta_1$ y $A \longrightarrow \alpha\beta_2$ con $\alpha \neq \epsilon$ y $\beta_1 \neq \beta_2$

- Estas gramáticas no son $LL(1)$
- según la longitud de α tampoco son $LL(k)$
- Notad que estar factorizada no garantiza $LL(1)$, ya que puede ser indirecto: $A \Longrightarrow^+_{iz} \alpha\beta_1$ y $A \Longrightarrow^+_{iz} \alpha\beta_2$ (se pueden expandir la reglas hasta que aflore, pero la gramática puede crecer mucho)

Transformación:
Reemplazamos $A \longrightarrow \alpha\beta_1 \mid \ldots \mid \alpha\beta_n \mid \delta_1 \mid \ldots \mid \delta_m$ por

$A \longrightarrow \alpha A' \mid \delta_1 \mid \ldots \mid \delta_m$ y $A' \longrightarrow \beta_1 \mid \ldots \mid \beta_n$

hasta que esté factorizada por la izquierda

Factorización por la izquierda

Consideramos la gramática no factorizada por la izquierda:

$$\left\{ \begin{array}{l} E \longrightarrow T + E \ \mid \ T \\ T \longrightarrow F * T \ \mid \ F \\ F \longrightarrow \underline{(}\ E\ \underline{)} \ \mid \ \textbf{id} \end{array} \right.$$

al factorizar por la izquierda obtenemos

Factorización por la izquierda

Consideramos la gramática no factorizada por la izquierda:

$$\begin{cases} E \longrightarrow T + E \mid T \\ T \longrightarrow F * T \mid F \\ F \longrightarrow \underline{(}\ E\ \underline{)} \mid \mathbf{id} \end{cases}$$

al factorizar por la izquierda obtenemos

$$\begin{cases} E \longrightarrow TE' \qquad\qquad E' \longrightarrow +E \mid \epsilon \\ \end{cases}$$

Factorización por la izquierda

Consideramos la gramática no factorizada por la izquierda:

$$\begin{cases} E \longrightarrow T + E \mid T \\ T \longrightarrow F * T \mid F \\ F \longrightarrow \underline{(}\ E\ \underline{)} \mid \textbf{id} \end{cases}$$

al factorizar por la izquierda obtenemos

$$\begin{cases} E \longrightarrow TE' & \qquad E' \longrightarrow +E \mid \epsilon \\ T \longrightarrow FT' & \qquad T' \longrightarrow *T \mid \epsilon \\ & \end{cases}$$

Factorización por la izquierda

Consideramos la gramática no factorizada por la izquierda:

$$\begin{cases} E \longrightarrow T + E \;\mid\; T \\ T \longrightarrow F * T \;\mid\; F \\ F \longrightarrow \underline{(}\; E \;\underline{)} \;\mid\; \textbf{id} \end{cases}$$

al factorizar por la izquierda obtenemos

$$\begin{cases} E \longrightarrow TE' & \qquad E' \longrightarrow +E \;\mid\; \epsilon \\ T \longrightarrow FT' & \qquad T' \longrightarrow *T \;\mid\; \epsilon \\ F \longrightarrow \underline{(}\; E \;\underline{)} \;\mid\; \textbf{id} & \end{cases}$$

Analizadores recursivos descendentes

Una alternativa para construir un analizador descendente para una gramática $LL(k)$:

- Para cada no-terminal A creamos una función f_A.
- Se elige la regla a aplicar usando $lookAhead(k)$ (versión eficiente) o se prueban todas las reglas hasta que una tiene éxito o todas fallan (versión ineficiente).
- Para cada $A \longrightarrow \alpha$ la función recorre α realizando las siguientes acciones:
  - Si encuentra un símbolo $a \in V_T$ y coincide con el siguiente en la entrada los consume, lo consume y continua. En caso contrario falla.
  - Si encuentra un símbolo $B \in V_N$ llama a la función f_B. Si f_B termina con éxito, continúa. En caso contrario falla.
  - Si consigue recorrer todo α sin fallo termina con éxito.
- Si la función f_S donde S es la raíz termina con éxito la frase es correcta. En caso contrario es que no lo es.

Contenidos

Especificación de expresiones

Para evitar la ambigüedad en la definición de expresiones (de cualquier tipo) con operadores unarios y binarios (infijos) debemos asociar **prioridades** y **asociatividades** a los operadores.

- Cada operador tiene un nivel de prioridad (suponemos que los niveles son consecutivos y comienzan en 0).
  En ausencia de paréntesis, los operadores más prioritarios se aplican antes que los menos prioritarios.
- Los operadores unarios pueden ser
  - asociativos: es posible escribir $- - - - 5$,
  - no asociativos: hay que utilizar paréntesis $-(-(-(-5)))$
- Los operadores binarios (infijos) pueden:
  - asociar a izquierdas: $a + a + a \equiv (a + a) + a$,
  - asociar a derechas: $a + a + a \equiv a + (a + a)$
  - no asociar: $a + a + a$ es sintácticamente incorrecto.

Prioridades y asociatividades

Para definir las prioridades y las asociatividades tenemos que:

- Introducimos un no terminal distinto para cada nivel de prioridad: $E_0, E_1, \ldots, E_n$
- En el nivel de prioridad i, para cada operador $\oplus$:
  - $\oplus$ es unario
    - asociativo prefijo: $E_i \longrightarrow \oplus E_i$
    - no asociativo prefijo: $E_i \longrightarrow \oplus E_{i+1}$
    - asociativo postfijo: $E_i \longrightarrow E_i \oplus$
    - no asociativo postfijo: $E_i \longrightarrow E_{i+1} \oplus$
  - $\oplus$ es binario (infijo)
    - asociativo a izquierdas: $E_i \longrightarrow E_i \oplus E_{i+1}$
    - asociativo a derechas: $E_i \longrightarrow E_{i+1} \oplus E_i$
    - no asociativo: $E_i \longrightarrow E_{i+1} \oplus E_{i+1}$
- En cada nivel se incluye una regla $E_i \longrightarrow E_{i+1}$
- Se añade un no terminal E_{n+1} con $E_{n+1} \longrightarrow num \mid id \mid \ldots \mid \underline{(}E_0\underline{)}$ para expresiones básicas o parentización

Prioridades y asociatividades

Para evitar la ambigüedad:

- un operador con la misma aridad no puede aparecer en más de un nivel de prioridad.
- En cada nivel de prioridad:
  - No puede haber operadores unarios prefijos asociativos y unarios posfijos asociativos.
  - No puede haber operadores unarios prefijos asociativos y binarios que asocien a izquierdas.
  - No puede haber operadores unarios posfijos asociativos y binarios que asocien a derechas.
  - No puede haber operadores binarios que asocien a izquierdas y binarios que asocien a derechas.

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$E_0 \longrightarrow$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$E_0 \longrightarrow * \; E_0$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$$E_0 \longrightarrow * E_0 \mid E_1 \,\&$$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$$E_0 \longrightarrow * \; E_0 \;\mid\; E_1 \;\&\; \mid\; - \; E_1$$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$$E_0 \longrightarrow * E_0 \mid E_1 \ \& \mid - E_1 \mid E_1$$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$E_0 \longrightarrow * \; E_0 \;\mid\; E_1 \; \& \;\mid\; - \; E_1 \;\mid\; E_1$

$E_1 \longrightarrow$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$E_0 \longrightarrow * E_0 \mid E_1 \,\& \mid - E_1 \mid E_1$

$E_1 \longrightarrow E_2 \,\&\, E_1$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$E_0 \longrightarrow * \, E_0 \;|\; E_1 \,\&\; |\; - \, E_1 \;|\; E_1$

$E_1 \longrightarrow E_2 \,\&\, E_1 \;|\; E_2 \,\%\, E_2$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$$E_0 \longrightarrow * \, E_0 \;\mid\; E_1 \,\&\; \mid\; - \, E_1 \;\mid\; E_1$$
$$E_1 \longrightarrow E_2 \,\&\, E_1 \;\mid\; E_2 \,\%\, E_2 \;\mid\; E_2$$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$E_0 \longrightarrow * \, E_0 \mid E_1 \,\&\, \mid - \, E_1 \mid E_1$

$E_1 \longrightarrow E_2 \,\&\, E_1 \mid E_2 \,\%\, E_2 \mid E_2$

$E_2 \longrightarrow$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$$E_0 \longrightarrow * \, E_0 \;\mid\; E_1 \,\&\; \mid\; -\, E_1 \;\mid\; E_1$$
$$E_1 \longrightarrow E_2 \,\&\, E_1 \;\mid\; E_2 \,\%\, E_2 \;\mid\; E_2$$
$$E_2 \longrightarrow E_2 \,!\, E_3$$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$$E_0 \longrightarrow * \, E_0 \;|\; E_1 \,\&\; |\; -\, E_1 \;|\; E_1$$
$$E_1 \longrightarrow E_2 \,\&\, E_1 \;|\; E_2 \,\%\, E_2 \;|\; E_2$$
$$E_2 \longrightarrow E_2 \,!\, E_3 \;|\; E_3$$

Prioridades y asociatividades. Ejemplo

Diseñar una gramática para un lenguaje de expresiones cuyas expresiones básicas son números, identificadores o expresiones con paréntesis y que involucra operadores con las siguientes características:

Operador	Tipo	Prioridad	Asociatividad
$*$	Unario prefijo	0	Asociativo
&	Unario postfijo	0	No asociativo
$-$	Unario prefijo	0	No asociativo
&	Binario infijo	1	Asociativo a derechas
%	Binario infijo	1	No asociativo
!	Binario infijo	2	Asociativo a izquierdas

Solución:

$$E_0 \longrightarrow * \ E_0 \ | \ E_1 \ \& \ | \ - \ E_1 \ | \ E_1$$
$$E_1 \longrightarrow E_2 \ \& \ E_1 \ | \ E_2 \ \% \ E_2 \ | \ E_2$$
$$E_2 \longrightarrow E_2 \ ! \ E_3 \ | \ E_3$$
$$E_3 \longrightarrow num \ | \ id \ | \ \underline{(} E_0 \underline{)}$$

Especificación de secuencias

- Utilizar recursión a izquierdas:
  - si usamos un reconocedor ascendente, es más eficiente;
  - si usamos un reconocedor descendente, transformamos la regla.

- Para especificar una secuencia de elementos con un separador ϕ: equivalente a expresiones con ϕ como operador infijo asociativo a izquierdas.

  Ejemplo: secuencia de instrucciones separadas por ";"
  $Ins \longrightarrow Ins\ ;\ I \mid I$

- Para una secuencia de elementos con un terminador ϕ:
  $Ins \longrightarrow Ins\ C \mid C$
  $C \longrightarrow I\ \phi$

  $Ins \longrightarrow Ins\ C \mid \epsilon$ si admitimos la lista vacía.

Tratamiento de la ambigüedad if–then-else

Supongamos que tenemos:

$S \longrightarrow SAsig \mid SWhile \mid \ldots \mid SIf$

y especificamos SIf de la siguiente forma:

$SIf \longrightarrow \underline{if}\ E\ \underline{then}\ S \mid \underline{if}\ E\ \underline{then}\ S\ \underline{else}\ S$

Entonces, la gramática resultante es ambigua, ya que

if $x = 5$ then if $y = 6$ then $x := x + 1$ else $x := x - 1$

tiene dos posibles interpretaciones según a quién asociemos el *else*

La forma de evitarlo por gramática es, de nuevo, definir niveles.

Tratamiento de la ambigüedad if–then-else

- Si se desea, como es usual, que el else siempre esté asociado a los if más internos:

$$\begin{array}{lcl} S0 & \longrightarrow & S0_{NB} \mid S1 \\ S1 & \longrightarrow & \underline{if}\ E\ \underline{then}\ S1\ \underline{else}\ S1 \mid S2 \\ S2 & \longrightarrow & SAsig \mid SWhile \mid \ldots \text{(todas las sentencias menos el if)} \end{array}$$

$$S0_{NB} \longrightarrow \underline{if}\ E\ \underline{then}\ S0_{NB} \mid \underline{if}\ E\ \underline{then}\ S1 \mid \underline{if}\ E\ \underline{then}\ S1\ \underline{else}\ S0_{NB}$$

$S0_{NB}$: representa las sentencias que contienen algún if sin parte else

La gramática es $LR(1)$ y $LALR(1)$. Puede transformarse a una gramática $LL(1)$ equivalente.

Tratamiento de la ambigüedad if–then-else

- Si se desea que el else siempre esté asociado a los if más externos:

$$\begin{array}{lll} S0 & \longrightarrow & \underline{if}\ E\ \underline{then}\ S0\ \underline{else}\ S0 \mid S1 \\ S1 & \longrightarrow & \underline{if}\ E\ \underline{then}\ S1 \mid S2 \\ S2 & \longrightarrow & SAsig \mid SWhile \mid \ldots \text{(todas las sentencias menos el if)} \end{array}$$

La gramática no es $LR(1)$! para ningún k. Cómo saber con $lookahead(k)$ si se han acabado los ifs con parte else?

- En la práctica en muchos casos se admite la ambigüedad y se resuelve a nivel de implementación, dando prioridad a la regla con *else*.

- Otra opción es explicitar claramente el final del if:

$$SIf \longrightarrow \underline{if}\ E\ \underline{then}\ S\ \underline{endif} \mid \underline{if}\ E\ \underline{then}\ S\ \underline{else}\ S\ \underline{endif}$$