Análisis Sintáctico Ascendente

CI4721 – Lenguajes de Programación II

Ernesto Hernández-Novich
<emhn@usb.ve>

Universidad "Simón Bolívar"

Copyright ©2012-2016

UNIVERSIDAD SIMÓN BOLÍVAR

Reconocedor ascendente determinístico

Sea $G = (N, \Sigma, P, S)$ una CFG cualquiera y $M = (Q, N \cup \Sigma, \delta_c, q_c, Q)$ su máquina característica determinística $LR(k)$. Entonces el PDA extendido

$$LR_k(G) = (\{s_0, s_1, s_2\}, \Sigma, Q, \delta, s_0, \{s_2\})$$

con δ definida según

$$\begin{aligned}
\delta(s_0, \lambda, \mathbf{u}, \lambda) &= \{(s_1, q_c)\} \\
\delta(s_1, \mathbf{a}, \mathbf{u}, q) &= \{(s_1, q'q)\}, \\
&\quad \text{si } [A \to \alpha \cdot \mathbf{a}\beta, w] \in q \wedge \mathbf{u} \in FIRST_k(\mathbf{a}\beta w) \wedge \delta_c(q, \mathbf{a}) = q' \\
\delta(s_1, \lambda, \mathbf{u}, q_0 q_1 \dots q_n) &= \{(s_1, q' q_n)\}, \\
&\quad \text{si } [A \to \alpha\cdot, u] \in q_0 \wedge |\alpha| = n \wedge A \neq S \wedge \delta_c(q_n, A) = q' \\
\delta(s_1, \lambda, \#^k, q_0 q_1 \dots q_n) &= \{(s_2, \lambda)\}, \\
&\quad \text{si } [S \to \alpha\cdot, \#^k] \in q_0 \wedge |\alpha| = n \wedge q_n = q_c
\end{aligned}$$

es un reconocedor ascendente y determinístico para G.

UNIVERSIDAD SIMÓN BOLÍVAR

¿Cómo convertirlo en un programa?

Convirtiendo transiciones en datos

- En nuestro PDA determinístico con *lookahead* . . .
  - El estado inicial solamente se usa para empilar q_c.
  - El estado final solamente acepta ante el final de la entrada.
  - El estado intermedio solamente tiene transiciones a sí mismo
    - Avanza sobre prefijos viables, recordando su contexto en la pila (*shift*).
    - Alcanzado un item completo, retorna la pila al contexto previo (*reduce*).

UNIVERSIDAD SIMÓN BOLÍVAR

¿Cómo convertirlo en un programa?

Convirtiendo transiciones en datos

- En nuestro PDA determinístico con *lookahead* . . .
  - El estado inicial solamente se usa para empilar q_c.
  - El estado final solamente acepta ante el final de la entrada.
  - El estado intermedio solamente tiene transiciones a sí mismo
    - Avanza sobre prefijos viables, recordando su contexto en la pila (*shift*).
    - Alcanzado un item completo, retorna la pila al contexto previo (*reduce*).
- Podemos diseñar un algoritmo muy compacto alrededor de
  - Una entrada con su marcador final $\#^k$.
  - Un *buffer* de tamaño k para el *lookahead*.
  - Una pila que comienza conteniendo q_c sobre un centinela $\#$.
  - Una tabla cuyas entradas indiquen para cada combinación de *lookahead* y tope de pila.
    - El estado a empilar, si se trata de un *shift*.
    - La regla a utilizar, si se trata de un *reduce*, junto al contexto al cual regresar una vez limpia la pila.

El algoritmo es independiente de la gramática solamente necesita la tabla.

UNIVERSIDAD SIMÓN BOLÍVAR

Reconocedor $LR(k)$ por Tabla

Inicialización

input: $w \in \Sigma$ con $\#^k$ marcadores y las dos partes de la Tabla M asociada a la gramática:

- *ACTION* – estado a empilar (*shift*) o regla a reducir (*reduce*).
- *GOTO* – ¿cómo regresar al punto que ocasionó la última reducción?

{Empilar el centinela y el símbolo inicial}
push$(\#)$
push(q_c)
{Preparar el $k-$*lookahead*}
$a \leftarrow$ los primeros k símbolos de w

UNIVERSIDAD SIMÓN BOLÍVAR

Reconocedor $LR(k)$ por Tabla

Procesamiento

while true do
  $s \leftarrow$ **top**()
  if $ACTION[s, a] = shift\ t$ **then**
    push(t) y consumir siguiente de w al final de a
  else if $ACTION[s, a] = reduce\ A \rightarrow \alpha$ **then**
    pop() tantas veces como $|\alpha|$
    $t \leftarrow$ **top**()
    push($GOTO[t, A]$)
    print $A \rightarrow \alpha$
  else if $ACTION[s, a] = accept$ **then**
    break
  else
    error
  end if
end while

output: Si $w \in L(G)$, la derivación más derecha, sino error

UNIVERSIDAD SIMÓN BOLÍVAR

El secreto está en la tabla

Características

- Una fila por cada $q \in Q$ de $M = (Q, N \cup \Sigma, \delta_c, q_c, Q)$
- $ACTION[q, u]$ una columna por cada *lookahead* posible – en cada posición se almacena
  - *shift* t – donde $t \in Q$
  - *reduce* $A \rightarrow \alpha$
  - *accept*
  - Un indicador de error sintáctico.
- $GOTO[q, A]$ tiene una columna por cada $N - \{S\}$ – en cada posición se almacena un $q \in Q$.

Los requerimientos de espacio son prohibitivos para $k > 1$

UNIVERSIDAD SIMÓN BOLÍVAR

Método $LR(0)$

El método más simple (e inútil en la vida real).

1. Se enumeran las producciones comenzando por cero.

Método $LR(0)$

El método más simple (e inútil en la vida real).

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $LR(0)$

El método más simple (e inútil en la vida real).

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $LR(0)$

El método más simple (e inútil en la vida real).

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in \Sigma \cup \{\#\}, ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$

UNIVERSIDAD SIMÓN BOLÍVAR

Método $LR(0)$

El método más simple (e inútil en la vida real).

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in \Sigma \cup \{\#\}, ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$
5. Cuando $[S \to \alpha\cdot] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $LR(0)$

El método más simple (e inútil en la vida real).

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in \Sigma \cup \{\#\}, ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$
5. Cuando $[S \to \alpha\cdot] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.
6. Cuando $[A \to \alpha \cdot B\beta] \in I_i \wedge \delta(I_i, B) = I_j$, entonces $GOTO[I_i, B] = I_j$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $LR(0)$

El método más simple (e inútil en la vida real).

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] =$ **shift** j.
4. Cuando $[A \to \alpha\cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in \Sigma \cup \{\#\}, ACTION[I_i, \mathbf{a}] =$ **reduce** p
5. Cuando $[S \to \alpha\cdot] \in I_i$, entonces $ACTION[I_i, \#] =$ **accept**.
6. Cuando $[A \to \alpha \cdot B\beta] \in I_i \wedge \delta(I_i, B) = I_j$, entonces $GOTO[I_i, B] = I_j$.
7. El resto de las entradas de $ACTION$ corresponde a **error**.

Si la gramática es $LR(0)$ todas las posiciones de $ACTION$ tendrán exactamente **una** acción.

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas

Paso 1 – Enumerar las producciones

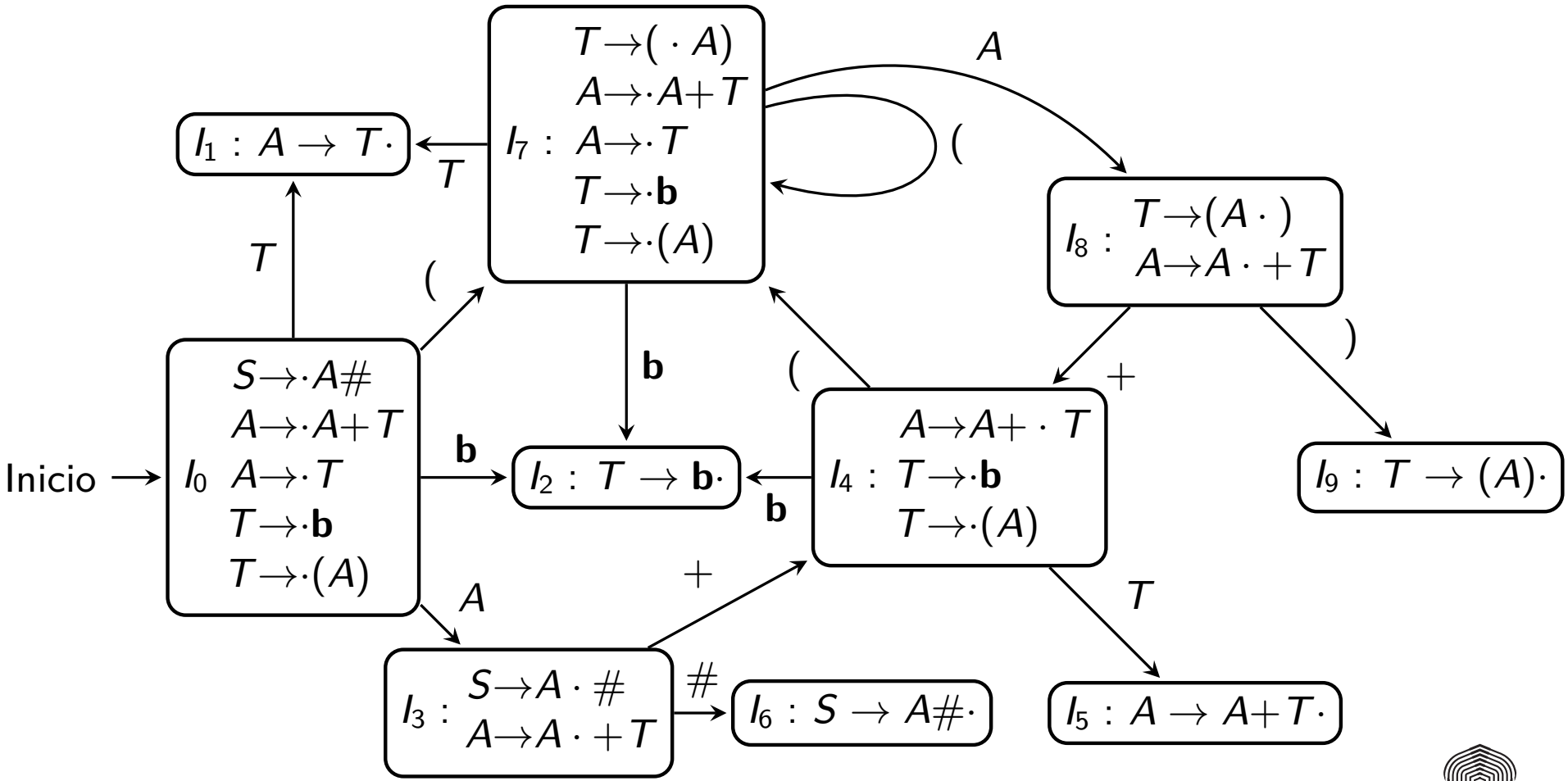
$$
\begin{array}{ll}
(0) & S \rightarrow A\# \\
(1) & A \rightarrow A + T \\
(2) & A \rightarrow T \\
(3) & T \rightarrow \mathbf{b} \\
(4) & T \rightarrow (A)
\end{array}
$$

- El símbolo inicial no es recursivo – la producción agregada siempre es la número cero.
- Se acostumbra abreviar **reduce** por **r** seguida del número de la regla.

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas

Paso 2 – Calcular la máquina de prefijos viables



UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas

Tablas *ACTION* y *GOTO* vacías

	b	+	(	)	#	A	T
I_0							
I_1							
I_2							
I_3							
I_4							
I_5							
I_6							
I_7							
I_8							
I_9							

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas

Paso 3 – Determinar los *shift*

	b	+	(	)	#	A	T
I_0	s2		s7				
I_1							
I_2							
I_3		s4			s6		
I_4	s2		s7				
I_5							
I_6							
I_7	s2		s7				
I_8		s4		s9			
I_9							

Expresiones aditivas

Paso 4 – Determinar los *reduce*

	b	+	(	)	#	A	T
I_0	s2		s7				
I_1	r2	r2	r2	r2	r2		
I_2	r3	r3	r3	r3	r3		
I_3		s4			s6		
I_4	s2		s7				
I_5	r1	r1	r1	r1	r1		
I_6							
I_7	s2		s7				
I_8		s4		s9			
I_9	r4	r4	r4	r4	r4		

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas

Paso 5 – Establecer el *accept*

	b	+	(	)	#	A	T
I_0	s2		s7				
I_1	r2	r2	r2	r2	r2		
I_2	r3	r3	r3	r3	r3		
I_3		s4			s6		
I_4	s2		s7				
I_5	r1	r1	r1	r1	r1		
I_6					acc		
I_7	s2		s7				
I_8		s4		s9			
I_9	r4	r4	r4	r4	r4		

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas

Paso 6 – Determinar los *goto*

	b	+	(	)	#	A	T
I_0	s2		s7			3	1
I_1	r2	r2	r2	r2	r2		
I_2	r3	r3	r3	r3	r3		
I_3		s4			s6		
I_4	s2		s7				5
I_5	r1	r1	r1	r1	r1		
I_6					acc		
I_7	s2		s7			8	1
I_8		s4		s9			
I_9	r4	r4	r4	r4	r4		

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas

Tabla $LR(0)$ terminada

	b	+	(	)	#	A	T
I_0	s2		s7			3	1
I_1	r2	r2	r2	r2	r2		
I_2	r3	r3	r3	r3	r3		
I_3		s4			s6		
I_4	s2		s7				5
I_5	r1	r1	r1	r1	r1		
I_6					acc		
I_7	s2		s7			8	1
I_8		s4		s9			
I_9	r4	r4	r4	r4	r4		

No hay conflictos – La gramática es $LR(0)$

UNIVERSIDAD SIMÓN BOLÍVAR

Método *SLR*(1)

Uso inocente del *lookahead*

1. Se enumeran las producciones comenzando por cero.

Método $SLR(1)$

Uso inocente del *lookahead*

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $SLR(1)$

Uso inocente del *lookahead*

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $SLR(1)$

Uso inocente del *lookahead*

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha \cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in FOLLOW_1(A), ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $SLR(1)$

Uso inocente del *lookahead*

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in FOLLOW_1(A), ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.
5. Cuando $[S \to \alpha\cdot] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $SLR(1)$

Uso inocente del *lookahead*

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in FOLLOW_1(A), ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.
5. Cuando $[S \to \alpha\cdot] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.
6. Cuando $[A \to \alpha \cdot B\beta] \in I_i \wedge \delta(I_i, B) = I_j$, entonces $GOTO[I_i, B] = I_j$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método $SLR(1)$

Uso inocente del *lookahead*

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(0)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $\forall \mathbf{a} \in FOLLOW_1(A), ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.
5. Cuando $[S \to \alpha\cdot] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.
6. Cuando $[A \to \alpha \cdot B\beta] \in I_i \wedge \delta(I_i, B) = I_j$, entonces $GOTO[I_i, B] = I_j$.
7. El resto de las entradas de $ACTION$ corresponde a **error**.

Si la gramática es $SLR(1)$ todas las posiciones de $ACTION$ tendrán exactamente **una** acción.

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas y multiplicativas

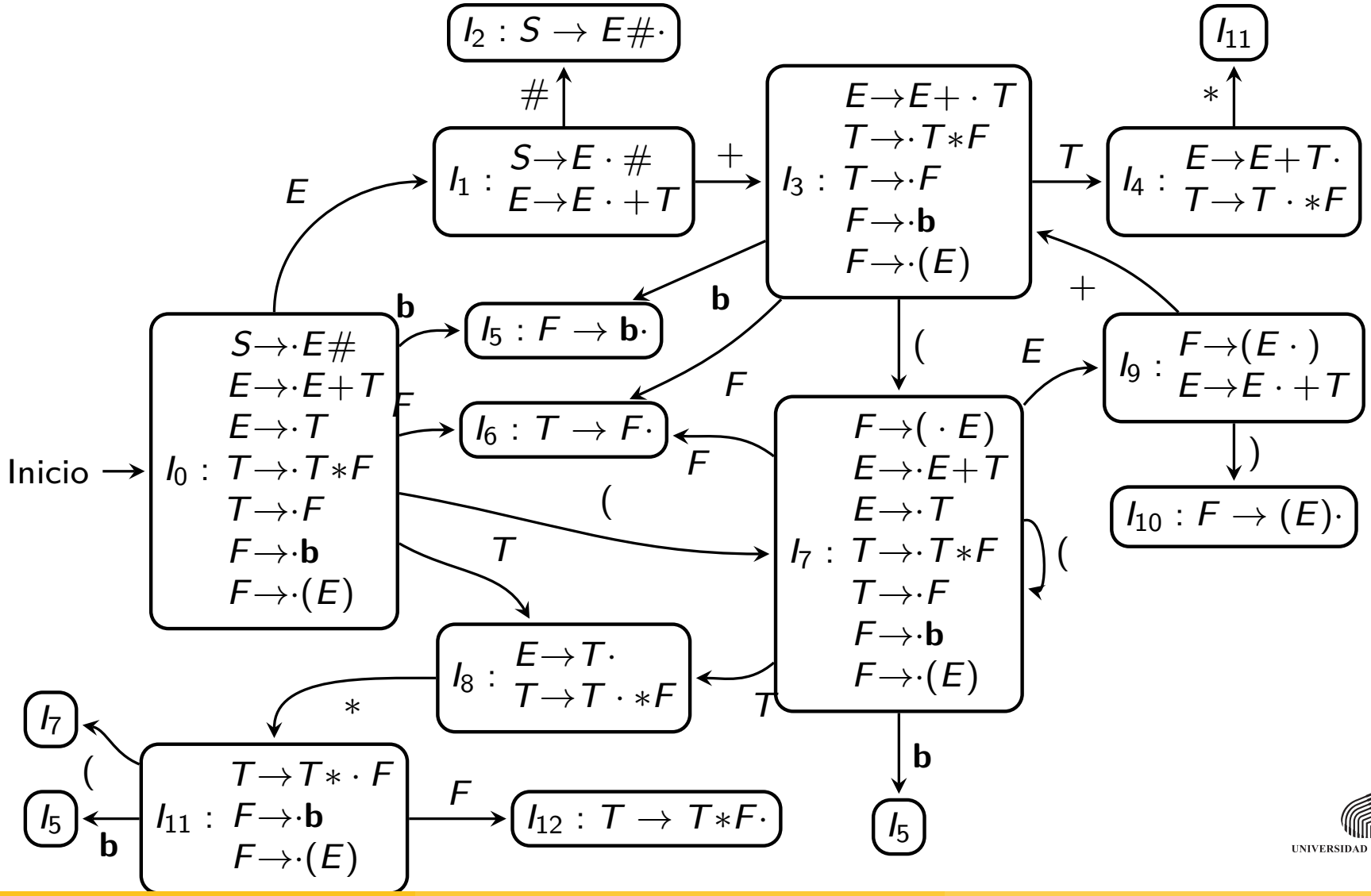
Paso 1 – Enumerar las producciones

$$
\begin{array}{ll}
(0) & S \rightarrow E\# \\
(1) & E \rightarrow E + T \\
(2) & E \rightarrow T \\
(3) & T \rightarrow T * F \\
(4) & T \rightarrow F \\
(5) & F \rightarrow \mathbf{b} \\
(6) & F \rightarrow (E)
\end{array}
$$

$$
\begin{aligned}
FOLLOW(S) &= \{\#\} \\
FOLLOW(E) &= \{\#, +,)\} \\
FOLLOW(T) &= \{\#, +,), *\} \\
FOLLOW(F) &= \{\#, +,), *\}
\end{aligned}
$$

Expresiones aditivas y multiplicativas

Paso 2 – Calcular la máquina de prefijos viables



Expresiones aditivas y multiplicativas

Tablas *ACTION* y *GOTO* vacías

	b	+	*	(	)	#	E	T	F
I_0									
I_1									
I_2									
I_3									
I_4									
I_5									
I_6									
I_7									
I_8									
I_9									
I_{10}									
I_{11}									
I_{12}									

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas y multiplicativas

Paso 3 – Determinar los *shift*

	b	+	*	(	)	#	E	T	F
I_0	s5			s7					
I_1		s3				s2			
I_2									
I_3	s5			s7					
I_4			s11						
I_5									
I_6									
I_7	s5			s7					
I_8			s11						
I_9		s3			s10				
I_{10}									
I_{11}	s5			s7					
I_{12}									

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas y multiplicativas

Paso 4 – Determinar los *reduce*

	b	+	*	(	)	#	E	T	F
I_0	s5			s7					
I_1		s3				s2			
I_2									
I_3	s5			s7					
I_4		r1	s11		r1	r1			
I_5		r5	r5		r5	r5			
I_6		r4	r4		r4	r4			
I_7	s5			s7					
I_8		r2	s11		r2	r2			
I_9		s3			s10				
I_{10}		r6	r6		r6	r6			
I_{11}	s5			s7					
I_{12}		r4	r4		r4	r4			

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas y multiplicativas

Paso 5 – Establecer el *accept*

	b	+	*	(	)	#	E	T	F
I_0	s5			s7					
I_1		s3				s2			
I_2						acc			
I_3	s5			s7					
I_4		r1	s11		r1	r1			
I_5		r5	r5		r5	r5			
I_6		r4	r4		r4	r4			
I_7	s5			s7					
I_8		r2	s11		r2	r2			
I_9		s3			s10				
I_{10}		r6	r6		r6	r6			
I_{11}	s5			s7					
I_{12}		r4	r4		r4	r4			

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas y multiplicativas

Paso 6 – Determinar los *goto*

	b	+	*	(	)	#	E	T	F
I_0	s5			s7			1	8	6
I_1		s3				s2			
I_2						acc			
I_3	s5			s7				4	6
I_4		r1	s11		r1	r1			
I_5		r5	r5		r5	r5			
I_6		r4	r4		r4	r4			
I_7	s5			s7			9	8	6
I_8		r2	s11		r2	r2			
I_9		s3			s10				
I_{10}		r6	r6		r6	r6			
I_{11}	s5			s7					12
I_{12}		r4	r4		r4	r4			

UNIVERSIDAD SIMÓN BOLÍVAR

Expresiones aditivas y multiplicativas

Tabla $SLR(1)$ terminada

	b	+	*	(	)	#	E	T	F
I_0	s5			s7			1	8	6
I_1		s3				s2			
I_2						acc			
I_3	s5			s7				4	6
I_4		r1	s11		r1	r1			
I_5		r5	r5		r5	r5			
I_6		r4	r4		r4	r4			
I_7	s5			s7			9	8	6
I_8		r2	s11		r2	r2			
I_9		s3			s10				
I_{10}		r6	r6		r6	r6			
I_{11}	s5			s7					12
I_{12}		r4	r4		r4	r4			

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

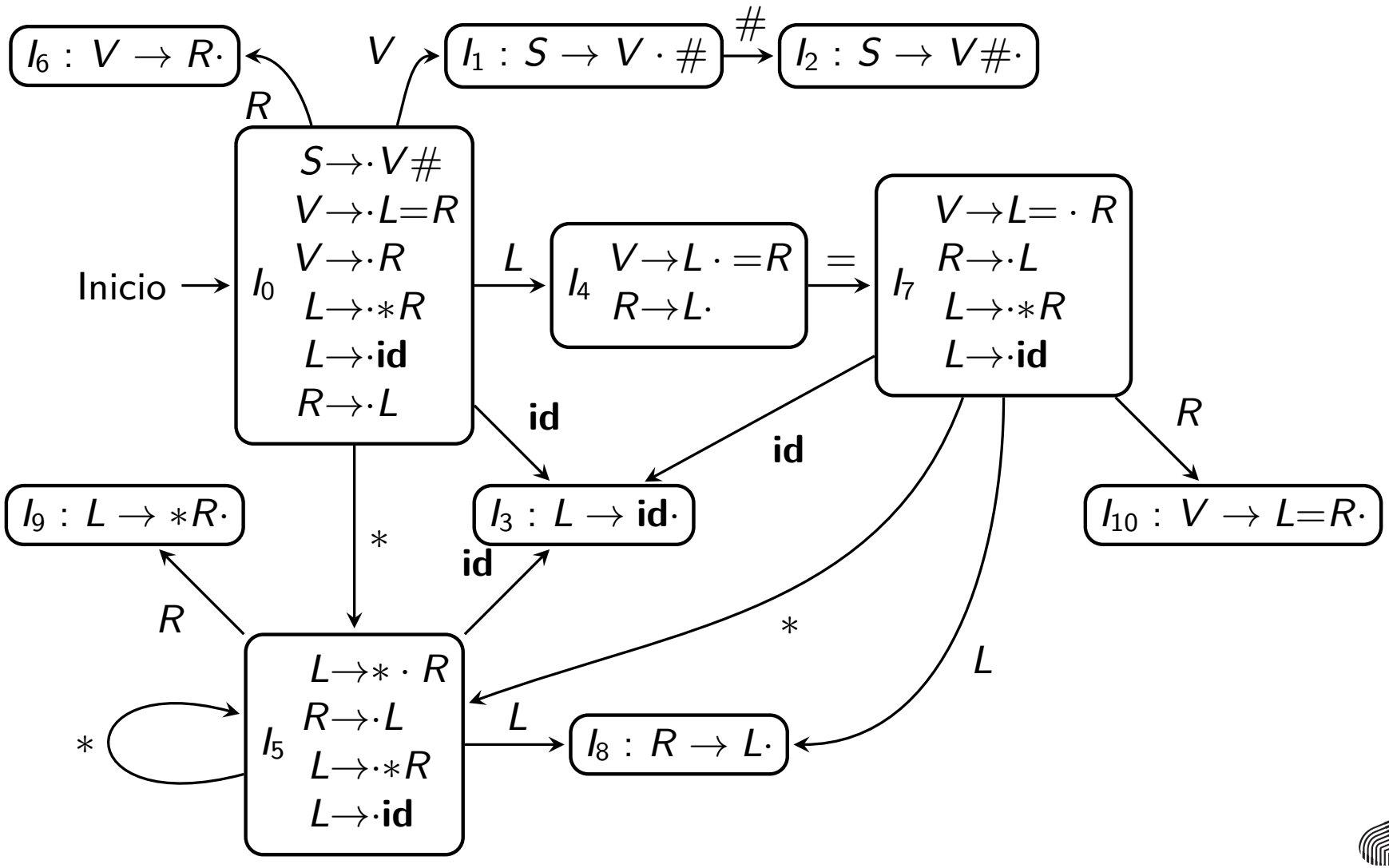
Paso 1 – Enumerar las producciones

$$
\begin{array}{ll}
(0) & S \rightarrow V\# \\
(1) & V \rightarrow L{=}R \\
(2) & V \rightarrow R \\
(3) & L \rightarrow {*}R \\
(4) & L \rightarrow \mathbf{id} \\
(4) & R \rightarrow L
\end{array}
$$

$$FOLLOW(S) = FOLLOW(V) = \{\#\}$$
$$FOLLOW(R) = FOLLOW(L) = \{\#, =\}$$

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores



UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Tablas *ACTION* y *GOTO* vacías

	id	=	*	#	V	L	R
I_0							
I_1							
I_2							
I_3							
I_4							
I_5							
I_6							
I_7							
I_8							
I_9							
I_{10}							

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Paso 3 – Determinar los *shift*

	id	=	*	#	V	L	R
I_0	s3		s5				
I_1				s2			
I_2							
I_3							
I_4		s7					
I_5	s3		s1				
I_6							
I_7	s3		s5				
I_8							
I_9							
I_{10}							

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Paso 4 – Determinar los *reduce*

	id	=	*	#	V	L	R
I_0	s3		s5				
I_1				s2			
I_2							
I_3	r4	r4	r4	r4			
I_4	r5	s7 /r5	r5	r5			
I_5	s3		s1				
I_6	r2	r2	r2	r2			
I_7	s3		s5				
I_8	r5	r5	r5	r5			
I_9	r3	r3	r3	r3			
I_{10}	r1	r1	r1	r1			

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Paso 5 – Establecer el *accept*

	id	=	*	#	V	L	R
I_0	s3		s5				
I_1				s2			
I_2				acc			
I_3	r4	r4	r4	r4			
I_4	r5	s7 /r5	r5	r5			
I_5	s3		s1				
I_6	r2	r2	r2	r2			
I_7	s3		s5				
I_8	r5	r5	r5	r5			
I_9	r3	r3	r3	r3			
I_{10}	r1	r1	r1	r1			

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Paso 6 – Determinar los *goto*

	id	=	*	#	V	L	R
I_0	s3		s5		1	4	6
I_1				s2			
I_2				acc			
I_3	r4	r4	r4	r4			
I_4	r5	s7 /r5	r5	r5			
I_5	s3		s1			8	9
I_6	r2	r2	r2	r2			
I_7	s3		s5			8	10
I_8	r5	r5	r5	r5			
I_9	r3	r3	r3	r3			
I_{10}	r1	r1	r1	r1			

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Tabla $LR(0)$ terminada

	id	=	*	#	V	L	R
I_0	s3		s5		1	4	6
I_1				s2			
I_2				acc			
I_3	r4	r4	r4	r4			
I_4	r5	s7 /r5	r5	r5			
I_5	s3		s1			8	9
I_6	r2	r2	r2	r2			
I_7	s3		s5			8	10
I_8	r5	r5	r5	r5			
I_9	r3	r3	r3	r3			
I_{10}	r1	r1	r1	r1			

No es $LR(0)$

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Tabla $SLR(1)$ terminada

	id	=	*	#	V	L	R
I_0	s3		s5		1	4	6
I_1				s2			
I_2				acc			
I_3		r4		r4			
I_4		s7 /r5		r5			
I_5	s3		s1			8	9
I_6				r2			
I_7	s3		s5			8	10
I_8		r5		r5			
I_9		r3		r3			
I_{10}		r1		r1			

No es $LR(0)$ ni $SLR(1)$ – el *FOLLOW* no es suficiente.

UNIVERSIDAD SIMÓN BOLÍVAR

Método Canónico $LR(1)$

El método más general – *lookahead* selectivo

1. Se enumeran las producciones comenzando por cero.

Método Canónico $LR(1)$

El método más general – *lookahead* selectivo

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(1)$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método Canónico $LR(1)$

El método más general – *lookahead* selectivo

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(1)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta, x] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método Canónico $LR(1)$

El método más general – *lookahead* selectivo

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(1)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta, x] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot, \mathbf{a}] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método Canónico $LR(1)$

El método más general – *lookahead* selectivo

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(1)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta, x] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot, \mathbf{a}] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.
5. Cuando $[S \to \alpha\cdot, \#] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método Canónico $LR(1)$

El método más general – *lookahead* selectivo

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(1)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta, x] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot, \mathbf{a}] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.
5. Cuando $[S \to \alpha\cdot, \#] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.
6. Cuando $[A \to \alpha \cdot B\beta, x] \in I_i \wedge \delta(I_i, B) = I_j$, entonces $GOTO[I_i, B] = I_j$.

UNIVERSIDAD SIMÓN BOLÍVAR

Método Canónico $LR(1)$

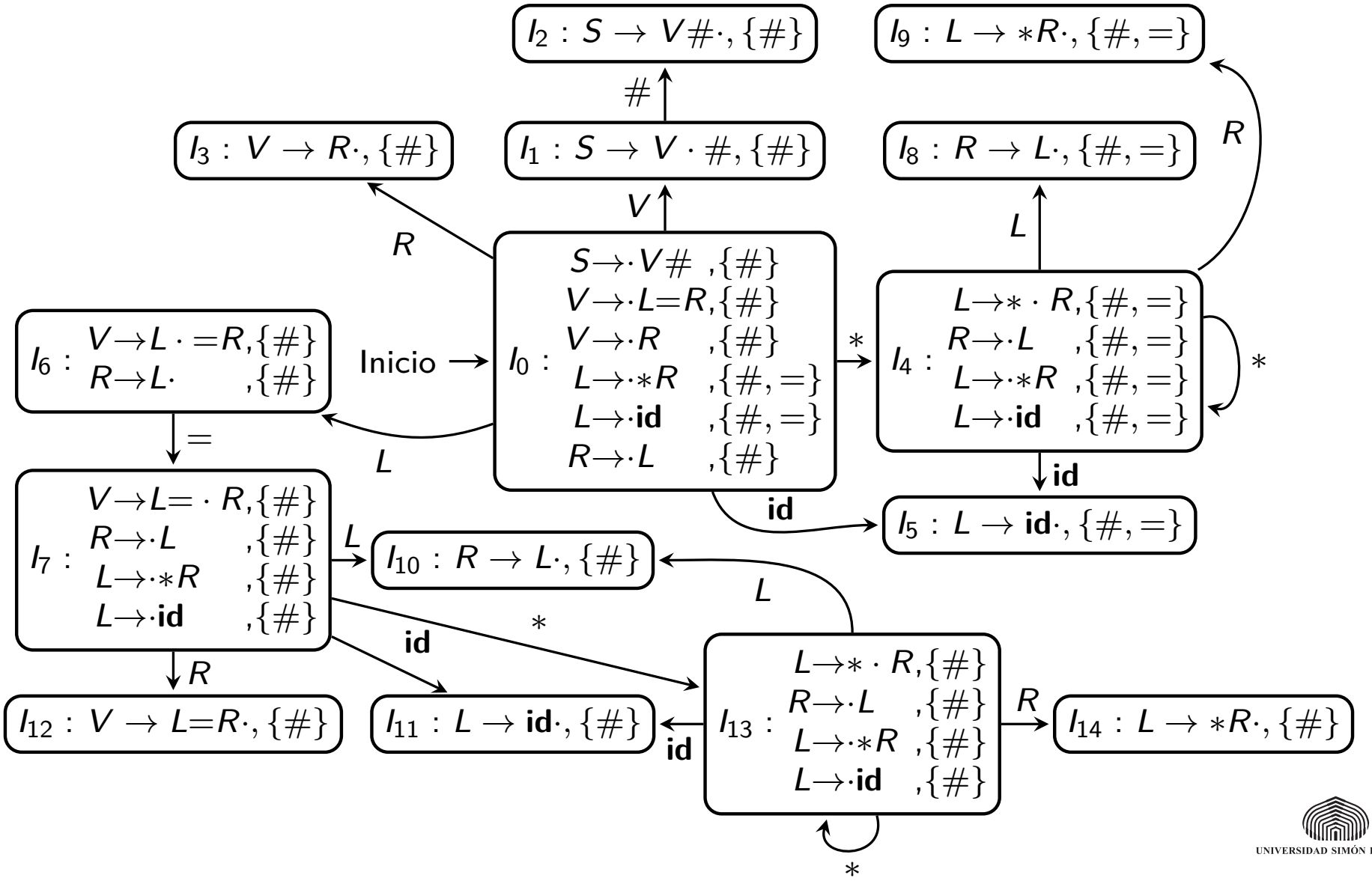
El método más general – *lookahead* selectivo

1. Se enumeran las producciones comenzando por cero.
2. Se construye la máquina característica determinística $LR(1)$.
3. Cuando $[A \to \alpha \cdot \mathbf{a}\beta, x] \in I_i \wedge \delta(I_i, \mathbf{a}) = I_j$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{shift } j$.
4. Cuando $[A \to \alpha\cdot, \mathbf{a}] \in I_i$ con $A \to \alpha$ la $p-$ésima producción y $A \neq S$, entonces $ACTION[I_i, \mathbf{a}] = \textbf{reduce } p$.
5. Cuando $[S \to \alpha\cdot, \#] \in I_i$, entonces $ACTION[I_i, \#] = \textbf{accept}$.
6. Cuando $[A \to \alpha \cdot B\beta, x] \in I_i \wedge \delta(I_i, B) = I_j$, entonces $GOTO[I_i, B] = I_j$.
7. El resto de las entradas de $ACTION$ corresponde a **error**.

Si la gramática es $LR(1)$ todas las posiciones de $ACTION$ tendrán exactamente **una** acción.

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores



UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Tablas *ACTION* y *GOTO* vacías

	id	=	*	#	V	L	R
I_0							
I_1							
I_2							
I_3							
I_4							
I_5							
I_6							
I_7							
I_8							
I_9							
I_{10}							
I_{11}							
I_{12}							
I_{13}							
I_{14}							

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Paso 3 – Determinar los *shift*

	id	=	*	#	V	L	R
I_0	s5		s4				
I_1				s2			
I_2							
I_3							
I_4	s5		s4				
I_5							
I_6		s7					
I_7	s11		s13				
I_8							
I_9							
I_{10}							
I_{11}							
I_{12}							
I_{13}	s11		s13				
I_{14}							

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Paso 4 – Determinar los *reduce*

	id	=	*	#	V	L	R
I_0	s5		s4				
I_1				s2			
I_2							
I_3				r2			
I_4	s5		s4				
I_5		r4		r4			
I_6		s7		r5			
I_7	s11		s13				
I_8		r5		r5			
I_9		r3		r3			
I_{10}				r5			
I_{11}				r4			
I_{12}				r1			
I_{13}	s11		s13				
I_{14}				r3			

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Paso 5 – Establecer el *accept*

	id	=	*	#	V	L	R
I_0	s5		s4				
I_1				s2			
I_2				acc			
I_3				r2			
I_4	s5		s4				
I_5		r4		r4			
I_6		s7		r5			
I_7	s11		s13				
I_8		r5		r5			
I_9		r3		r3			
I_{10}				r5			
I_{11}				r4			
I_{12}				r1			
I_{13}	s11		s13				
I_{14}				r3			

l-values y *r-values* con apuntadores

Paso 6 – Determinar los *goto*

	id	=	*	#	V	L	R
I_0	s5		s4		1	6	3
I_1				s2			
I_2				acc			
I_3				r2			
I_4	s5		s4			8	9
I_5		r4		r4			
I_6		s7		r5			
I_7	s11		s13			10	12
I_8		r5		r5			
I_9		r3		r3			
I_{10}				r5			
I_{11}				r4			
I_{12}				r1			
I_{13}	s11		s13			10	14
I_{14}				r3			

UNIVERSIDAD SIMÓN BOLÍVAR

l-values y *r-values* con apuntadores

Tabla Canónica $LR(1)$ terminada

	id	=	*	#	V	L	R
I_0	s5		s4		1	6	3
I_1				s2			
I_2				acc			
I_3				r2			
I_4	s5		s4			8	9
I_5		r4		r4			
I_6		s7		r5			
I_7	s11		s13			10	12
I_8		r5		r5			
I_9		r3		r3			
I_{10}				r5			
I_{11}				r4			
I_{12}				r1			
I_{13}	s11		s13			10	14
I_{14}				r3			

Es $LR(1)$

UNIVERSIDAD SIMÓN BOLÍVAR

Recuperación de Errores

. . . porque el mundo no es perfecto

- Un reconocedor ascendente detecta errores al consultar *ACTION*.
  - Canónico $LR(1)$ **jamás** hace reducciones antes de reportar el error.
  - $SLR(1)$ (y el $LALR(1)$ que no hemos estudiado) podrían hacer reducciones antes de reportar el error.
  - Ninguno ejecutará *shift* de un símbolo erróneo.
- Abortar el reconocimiento es inaceptable
  - El programa "está mal" – el proceso de síntesis no ocurrirá, pero el análisis debería continuar tanto como se pueda.
  - Cada error debe ser amplio y detallado
    - Ubicación – línea y columna, contexto de ser posible.
    - Condición – "esperaba X pero recibí Y"
  - Encontrar otros defectos ayudará al programador.
- Existen dos técnicas de recuperación aplicables

UNIVERSIDAD SIMÓN BOLÍVAR

Recuperación de Errores

. . . porque el mundo no es perfecto

- Un reconocedor ascendente detecta errores al consultar *ACTION*.
  - Canónico $LR(1)$ **jamás** hace reducciones antes de reportar el error.
  - $SLR(1)$ (y el $LALR(1)$ que no hemos estudiado) podrían hacer reducciones antes de reportar el error.
  - Ninguno ejecutará *shift* de un símbolo erróneo.
- Abortar el reconocimiento es inaceptable
  - El programa "está mal" – el proceso de síntesis no ocurrirá, pero el análisis debería continuar tanto como se pueda.
  - Cada error debe ser amplio y detallado
    - Ubicación – línea y columna, contexto de ser posible.
    - Condición – "esperaba X pero recibí Y"
  - Encontrar otros defectos ayudará al programador.
- Existen dos técnicas de recuperación aplicables
  - Técnica del Pánico (*Panic Mode*).
  - Técnica del Engaño (*Phrase Level Recovery*).

UNIVERSIDAD SIMÓN BOLÍVAR

Técnica del Pánico

Panic Mode, a.k.a. Discard all the tokens!

- Desempilar hasta encontrar I_a con $GOTO$ hacia algún no terminal A particular – retroceder en el prefijo que no se pudo completar.
- Descartar *tokens* hasta encontrar alguno que esté en $FOLLOW(A)$.
- Empila $GOTO(s, A)$ y continuar – simula una reducción exitosa.
- Usualmente A corresponde a un elemento sintáctico complejo – instrucción, expresión, bloque, ...
  - Si A corresponde a instrucción, entonces el token podría ser `;`.
  - La posición en la tabla apunta a conjuntos de sincronización – tuplas de no terminal y terminal asociado.

UNIVERSIDAD SIMÓN BOLÍVAR

Técnica del Engaño

(Phrase Level Recovery, a.k.a. Let me type for you)

- La posición de la tabla apunta a una *subrutina* de manejo del error.
- Cada rutina es específica para la recuperación particular – altamente dependiente del lenguaje.
- Las rutinas "completan" lo que falta para continuar
  - Agregan, quitan o cambian símbolos en la entrada.
  - Agregan o sacan cosas de la pila – Muy peligroso.
  - Anuncian lo que hicieron para "corregir" el problema.
- Solamente aplicable a lenguajes (o sub-lenguajes) tales que
  - Se conocen los errores más frecuentes.
  - Es fácil alcanzar formas sentenciales válidas con edición mínima.

UNIVERSIDAD SIMÓN BOLÍVAR

Técnica del Engaño

(Phrase Level Recovery, a.k.a. Let me type for you)

- La posición de la tabla apunta a una *subrutina* de manejo del error.
- Cada rutina es específica para la recuperación particular – altamente dependiente del lenguaje.
- Las rutinas "completan" lo que falta para continuar
  - Agregan, quitan o cambian símbolos en la entrada.
  - Agregan o sacan cosas de la pila – Muy peligroso.
  - Anuncian lo que hicieron para "corregir" el problema.
- Solamente aplicable a lenguajes (o sub-lenguajes) tales que
  - Se conocen los errores más frecuentes.
  - Es fácil alcanzar formas sentenciales válidas con edición mínima.

No la llame *ad hoc*, llámela *ad hack*.

UNIVERSIDAD SIMÓN BOLÍVAR

Consideraciones

- El método $LR(0)$ es demasiado limitado.

Consideraciones

- El método $LR(0)$ es demasiado limitado.
- El método $SLR(1)$ es suficiente para gramáticas simples
  - Se basa en las limitaciones del $LR(0)$.
  - Hace una aproximación "gruesa" usando $FOLLOW_1$.
  - Las gramáticas $SLR(1)$ no son ambiguas, pero hay gramáticas no ambiguas que no son $SLR(1)$

UNIVERSIDAD SIMÓN BOLÍVAR

Consideraciones

- El método $LR(0)$ es demasiado limitado.
- El método $SLR(1)$ es suficiente para gramáticas simples
  - Se basa en las limitaciones del $LR(0)$.
  - Hace una aproximación "gruesa" usando $FOLLOW_1$.
  - Las gramáticas $SLR(1)$ no son ambiguas, pero hay gramáticas no ambiguas que no son $SLR(1)$
- El método $LR(1)$ es el más general.
  - Toda gramática $LR(0)$ o $SLR(1)$ también es $LR(1)$.
  - La cantidad de estados aumenta notablemente – en nuestro ejemplo aumentó en un 37 % de $SLR(1)$ a $LR(1)$.
  - Tabla $GOTO$ es brutalmente esparcida – en nuestro ejemplo hay 80 % **vacío**.

UNIVERSIDAD SIMÓN BOLÍVAR

Consideraciones

- El método $LR(0)$ es demasiado limitado.
- El método $SLR(1)$ es suficiente para gramáticas simples
  - Se basa en las limitaciones del $LR(0)$.
  - Hace una aproximación "gruesa" usando $FOLLOW_1$.
  - Las gramáticas $SLR(1)$ no son ambiguas, pero hay gramáticas no ambiguas que no son $SLR(1)$
- El método $LR(1)$ es el más general.
  - Toda gramática $LR(0)$ o $SLR(1)$ también es $LR(1)$.
  - La cantidad de estados aumenta notablemente – en nuestro ejemplo aumentó en un 37 % de $SLR(1)$ a $LR(1)$.
  - Tabla $GOTO$ es brutalmente esparcida – en nuestro ejemplo hay 80 % **vacío**.

One does not simply build an $LR(1)$ parser manually.

UNIVERSIDAD SIMÓN BOLÍVAR

Bibliografía

- [*Aho*]
  - Secciones 4.6 y 4.7
  - Ejercicios 4.6.1 a 4.6.9
- Procese la gramática inicial de mini JSON
  - Construya las tablas $LR(0)$, $SLR(1)$ y $LR(1)$.
  - Compare sus tamaños y manejo de conflictos.

UNIVERSIDAD SIMÓN BOLÍVAR