Tema 5. Análisis sintáctico ascendente

1. Conceptos fundamentales
2. Análisis LR(0)
3. Análisis SLR(1)
4. Resolución de conflictos
5. Recuperación de errores
6. Relaciones entre gramáticas
7. Generadores de Analizadores Sintácticos (seminario lab.)

1. Conceptos fundamentales

ASA: Inversa de derivación a derechas

E-> E+T (1)
  | T (2)
T-> T * F (3)
  | F (4)
F-> (E) (5)
  | id (6)

ASA para cadena ω = id + id * id

Alg. ASA red–despl (con backtracking)

repetir

  repetir

    Si en la cima de la pila está el lado derecho de una producción REDUCIR

    Si no DESPLAZAR el símbolo de la cadena de entrada a la pila

  hasta no queden símbolos en la cadena de entrada

  Si la pila solo tiene el símbolo inicial de la cadena: ACEPTAR

  Si no: Aplicar vuelta atrás: Deshacer última reducción y probar con desplazamiento

hasta probar todas las posibilidades

Error

Alg. ASA red–desp (con backtracking)

Ejemplo

S-> E F

E-> a b

F-> a b a

ω= a b a b a

PILA	ENTRADA	SALIDA
\$	a b a b a \$	
\$ a	b a b a \$	
\$ a b	a b a \$	
\$ E	a b a \$	red-2
\$ E a	b a \$	
\$ E a b	a \$	
\$ E E	a \$	red-2
\$ E E a	\$	vuelta atrás
\$ E a b	a \$	
\$ E a b a	\$	
\$ E F	\$	red-3
\$ S	\$	red-1
ACEPTACIÓN		

Pivote

Dada una gramática incontextual $G=(N,\Sigma,P,S)$, y dada una forma sentencial a derechas $\alpha\ \beta\ \omega \in (N \cup \Sigma)^*$, un **pivote** de $\alpha\ \beta\ \omega$ es (r, j), donde $r \in P$ y $j \geq 0$, con $A \in N$; $\alpha, \beta \in (N \cup \Sigma)^*$; $\omega \in \Sigma^*$,

si cumple:

$$S \Rightarrow^* \alpha\ A\ \omega \Rightarrow \alpha\ \beta\ \omega$$

$$\text{con} \quad r = (A \rightarrow \beta) \ \text{ y } \ j = |\alpha\ \beta|$$

Gramática LR(k)

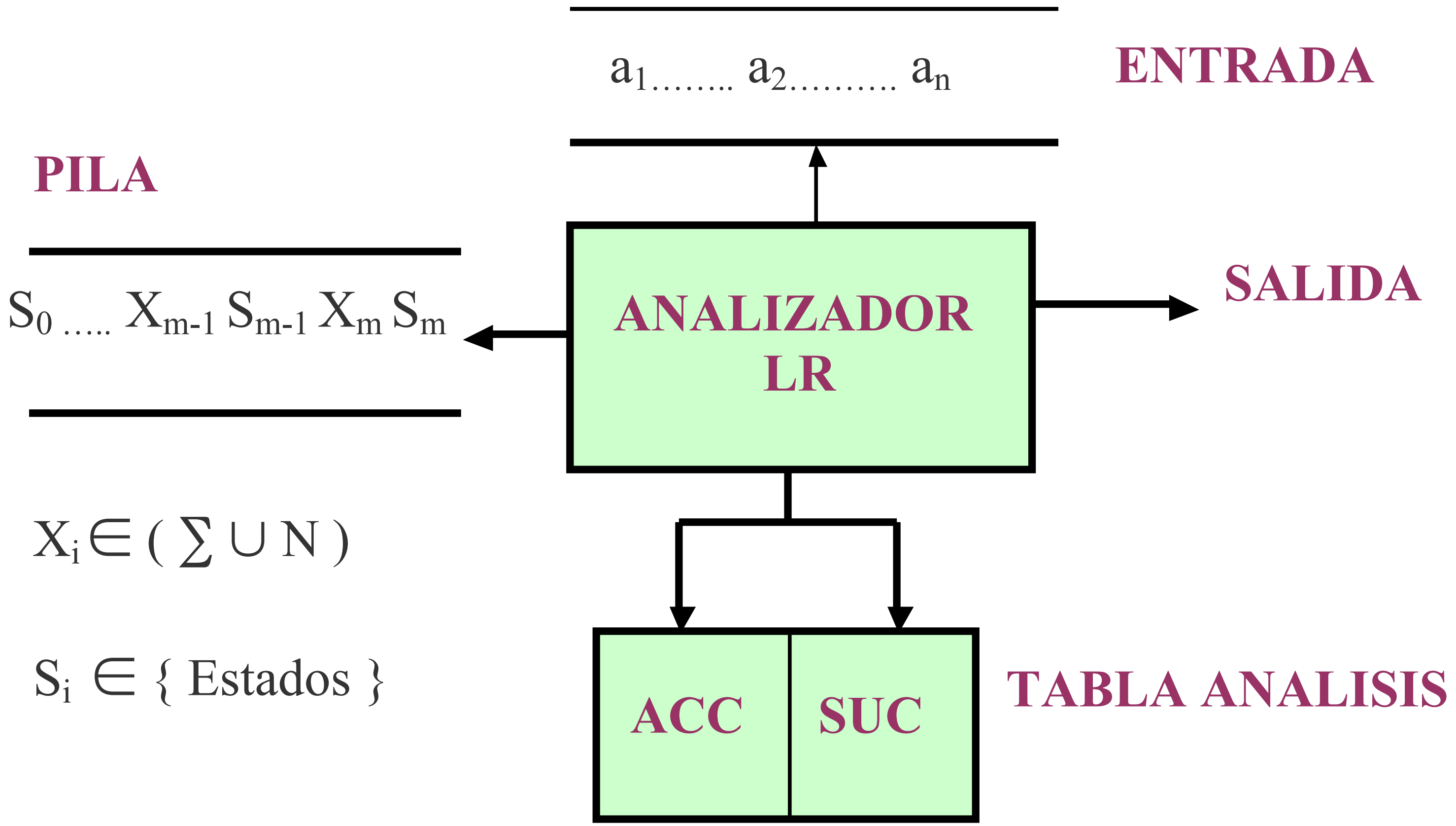
Dado un $k \geq 0$, y una gramática incontextual $G=(N,\Sigma,P,S)$ reducida (el axioma no aparece en la parte derecha de ninguna regla de G), G es **LR(k)** si para $\gamma, \alpha, \alpha', \beta, \beta' \in (N \cup \Sigma)^*$; $\omega, \omega' \in \Sigma^*$; $A, A' \in N$ se tiene que,

Si se cumple:

1. $S \Rightarrow^* \alpha\ A\ \omega \Rightarrow \alpha\ \beta\ \omega = \gamma\ \omega \quad (A \rightarrow \beta, |\alpha\ \beta|)$
2. $S \Rightarrow^* \alpha'\ A'\ \omega' \Rightarrow \alpha'\ \beta'\ \omega' = \gamma\ \omega' \quad (A' \rightarrow \beta', |\alpha'\ \beta'|)$
3. $\text{Primeros}_k\ (\omega) = \text{Primeros}_k\ (\omega')$

Entonces $(A \rightarrow \beta, |\alpha\ \beta|) = (A' \rightarrow \beta', |\alpha'\ \beta'|)$

Modelo de análisis LR



Configuración: $(S_0\, X_1\, S_1\, X_2 \ldots X_n S_n,\ \omega\ \$,\ \pi)$ *Configuración inicial:* $(S_0\ ,\ \omega\ \$,\ \varepsilon)$

acc: $Q \times (\Sigma \cup \{\$\})^k$ --> {des, red-r, err, accep}

suc: $Q \times (\Sigma \cup N)$ --> $Q \cup$ {err}

Algoritmo A.S.A.

ALGORITMO A.S.A:Desplazamiento-Reducción /* Tablas compactadas */

ENTRADA G' = (N', Σ, P', S'); $\omega \in \Sigma^*$; TA (acc,suc)

acc: Q x (Σ ∪ {$}) → {des-s, red-r, err, acep}

suc: Q x N → Q ∪ {err}

SALIDA Si $\omega \in L(G)$ entonces π else error( )

METODO apilar(s_0); sim := yylex(); $\pi := \varepsilon$; fin := falso;

```
repetir
    Caso acc[cima,sim] sea
      "des-s":   apilar(sim);   apilar(s);   sim := yylex();
      "red-r: A → β":  para i:=1 hasta 2*|β| hacer desapilar;
                       apilar(A);  apilar(suc[cima-1,A]);  π = π · r;
      "acep":  fin := verdad;
      "err":   yyerror( );
    end;
hasta fin
```

FIN

Ejemplo: Gramática G_1

E' -> E	(0)
E -> E **+** T	(1)
| T	(2)
T -> T ***** F	(3)
| F	(4)
F -> **(** E **)**	(5)
| **id**	(6)

Ejemplo (Gramática G_1)

Sin compactar

	ACC						SUCC							
	id	+	*	(	)	$	id	+	*	(	)	E	T	F
0	d			d			5			4		1	2	3
1		d				Ac		6						
2		r-2	d		r-2	r-2			7					
3		r-4	r-4		r-4	r-4								
4	d			d			5			4		8	2	3
5		r-6	r-6		r-6	r-6								
6	d			d			5			4			9	3
7	d			d			5			4				10
8		d			d			6			11			
9		r-1	d		r-1	r-1			7					
10		r-3	r-3		r-3	r-3								
11		r-5	r-5		r-5	r-5								

Ejemplo (Gramática G1)

Compactadas

	ACC						SUCC		
	id	+	*	(	)	$	E	T	F
0	d-5			d-4			1	2	3
1		d-6				Ac			
2		r-2	d-7		r-2	r-2			
3		r-4	r-4		r-4	r-4			
4	d-5			d-4			8	2	3
5		r-6	r-6		r-6	r-6			
6	d-5			d-4				9	3
7	d-5			d-4					10
8		d-6			d-11				
9		r-1	d-7		r-1	r-1			
10		r-3	r-3		r-3	r-3			
11		r-5	r-5		r-5	r-5			

Ejemplo (Gramática G1)

	id	+	*	(	)	$	E	T	F
0	d-5			d-4			1	2	3
1		d-6				Ac			
2		r-2	d-7		r-2	r-2			
3		r-4	r-4		r-4	r-4			
4	d-5			d-4			8	2	3
5		r-6	r-6		r-6	r-6			
6	d-5			d-4				9	3
7	d-5			d-4					10
8		d-6			d-11				
9		r-1	d-7		r-1	r-1			
10		r-3	r-3		r-3	r-3			
11		r-5	r-5		r-5	r-5			

E’ ::= E 0)

E ::= E **+** T (1)

E ::= T (2)

T ::= T ***** F (3)

T ::= F (4)

F ::= **(** E **)** (5)

F ::= **id** (6)

Traza ASA: (0, id+id*id$,)

Prefijo viable. Teorema de Knuth

Un **prefijo viable** para una forma sentencial a derechas $\alpha\ \beta\ \omega$ con α, $\beta \in (N \cup \Sigma)^*$; $\omega \in \Sigma^*$, siendo su pivote asociado $(A \rightarrow \beta, |\alpha\ \beta|)$ y $\alpha\ \beta = u_1 \ldots u_n$, es cualquier cadena $u_1 \ldots u_i$ con $1 \leq i \leq n$.

Teorema de Knuth

El conjunto de todos los prefijos viables de cualquier forma sentencial a derechas de una gramática LR(k), puede ser reconocido por un Autómata de Estados Finitos.

2. Análisis LR(0)

Elemento LR(0)

Sea G = (N,**S**,P,S) una gramática incontextual reducida.

Un elemento LR(o) para G es $[A \rightarrow \beta_1 \cdot \beta_2]$; siendo

$(A \rightarrow \beta_1\ \mathbf{b_2}) \in P$.

Nota. Para la producción $A \rightarrow \varepsilon$ el único elemento LR(o) se representa por $[A \rightarrow .]$

Un elemento LR(o) $[A \rightarrow \beta_1 \cdot \beta_2]$ es un elemento LR(o) válido para un prefijo viable $(\alpha\ \beta_1)$;

si dado α, β_1, $\beta_2 \in (N \cup \Sigma)^*$; $A \in N$; $\omega \in \Sigma^*$, cumple que:

$$S \Rightarrow^* \alpha\ A\ \omega \Rightarrow \alpha\ \beta_1\ \beta_2\ \omega$$

siendo el pivote $(A \rightarrow \beta_1\ \beta_2, |\alpha\ \beta_1\ \beta_2|)$

Elemento LR(0)

Teorema:

Si un elemento LR(o) $[A \rightarrow \beta_1 \cdot X\ \beta_2]$ es un elemento LR(o) válido para un prefijo viable **($\alpha\ \beta_1$) entonces**

$[A \rightarrow \beta_1 X\ \cdot \beta_2]$ es un elemento LR(o) válido para el prefijo viable **($\alpha\ \beta_1 X$)**

$X \in N \cup \Sigma$

Teorema:

Si un elemento LR(o) $[A \rightarrow \beta_1 \cdot B\ \beta_2]$ es un elemento LR(o) válido para un prefijo viable **($\alpha\ \beta_1$) entonces**

$[B \rightarrow\ \cdot \beta_3]$ es un elemento LR(o) válido para el prefijo viable **($\alpha\ \beta_1$)**

$B \in N \qquad (B \rightarrow \beta_3) \in P$

Algoritmos Clausura y Sucesor

Clausura (I)

clausura (I) := I;

Repetir

para todo $[A \rightarrow \alpha \cdot B\ \beta] \in$ clausura (I) hacer

para todo $(B \rightarrow \gamma) \in P$: $[B \rightarrow \cdot\ \gamma] \notin$ clausura (I) hacer

clausura (I) := clausura (I) $\cup$ $\{[B \rightarrow \cdot\ \gamma]\}$;

Hasta no se incorporen elementos LR(0) a clausura(I);

Sucesor (I, x)

J := $\varnothing$;

para todo $[A \rightarrow \mathbf{a} \cdot x\ \beta] \in I$

hacer $J = J \cup \{[A \rightarrow \mathbf{a}\ x \cdot \beta]\}$;

sucesor (I,x) := clausura (J)

Ejemplo: Gramática G_2

G_2 :

S -> A b (1)

A -> a A (2)

A -> b (3)

Algoritmo colección canónica de elementos LR(0)

ENTRADA $G' = (N', \Sigma, P', S')$;

SALIDA C: Colección canónica de conjuntos de elementos LR(0)

METODO

C = { clausura ({ [$S' \to \cdot$ S] }) };   /* Sin marcar */

Repetir

para todo I no marcado $\in$ C hacer

para todo x $\in$ ($N' \cup$ **S**) hacer

si sucesor(I, x) $\neq \varnothing \wedge$ sucesor(I, x) $\notin$ C entonces

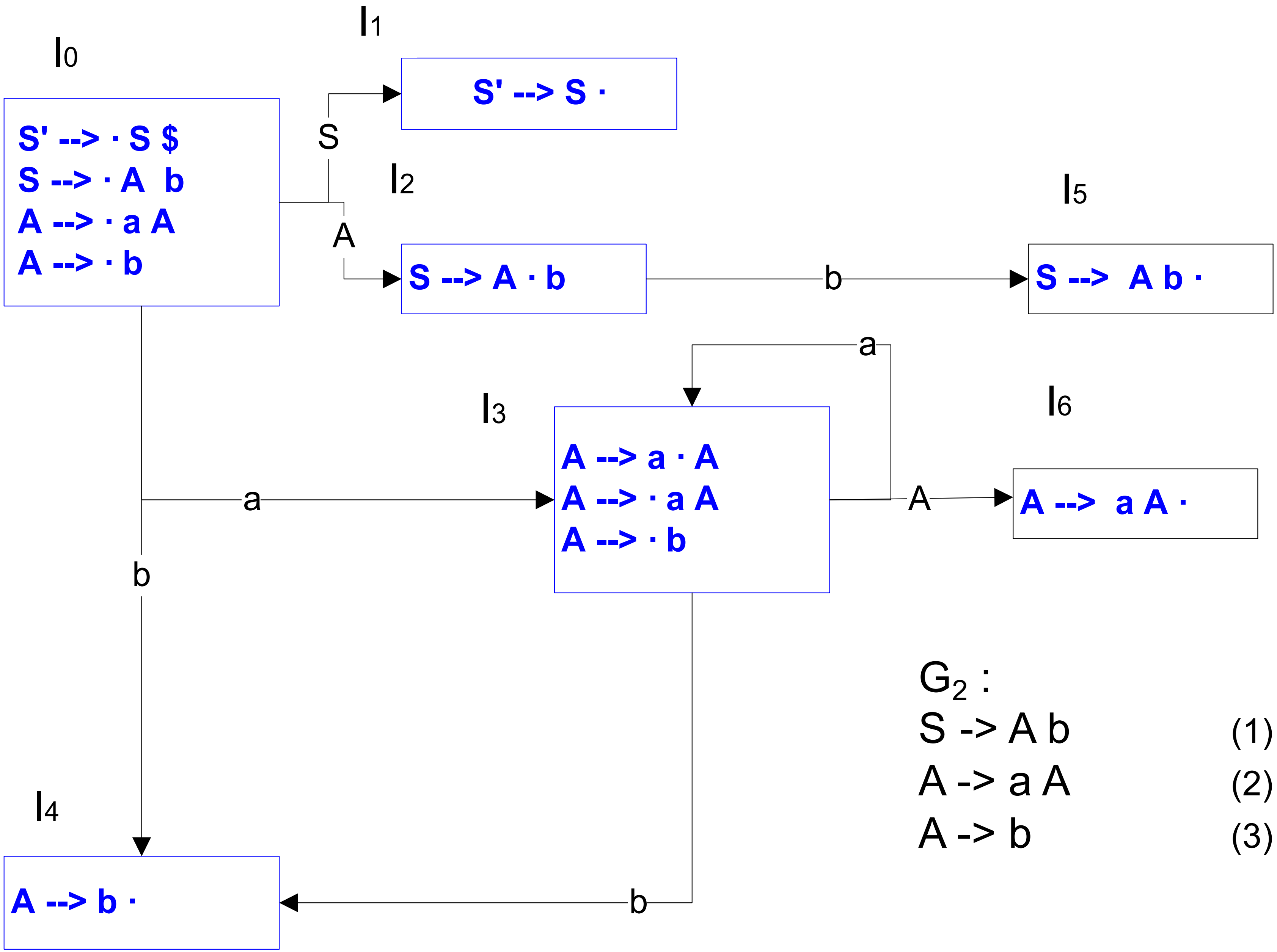
C := C $\cup$ sucesor (I, x) ;

marcar I

Hasta no queden en C conjuntos no marcados ;

FIN

Ejemplo: Gramática G_2



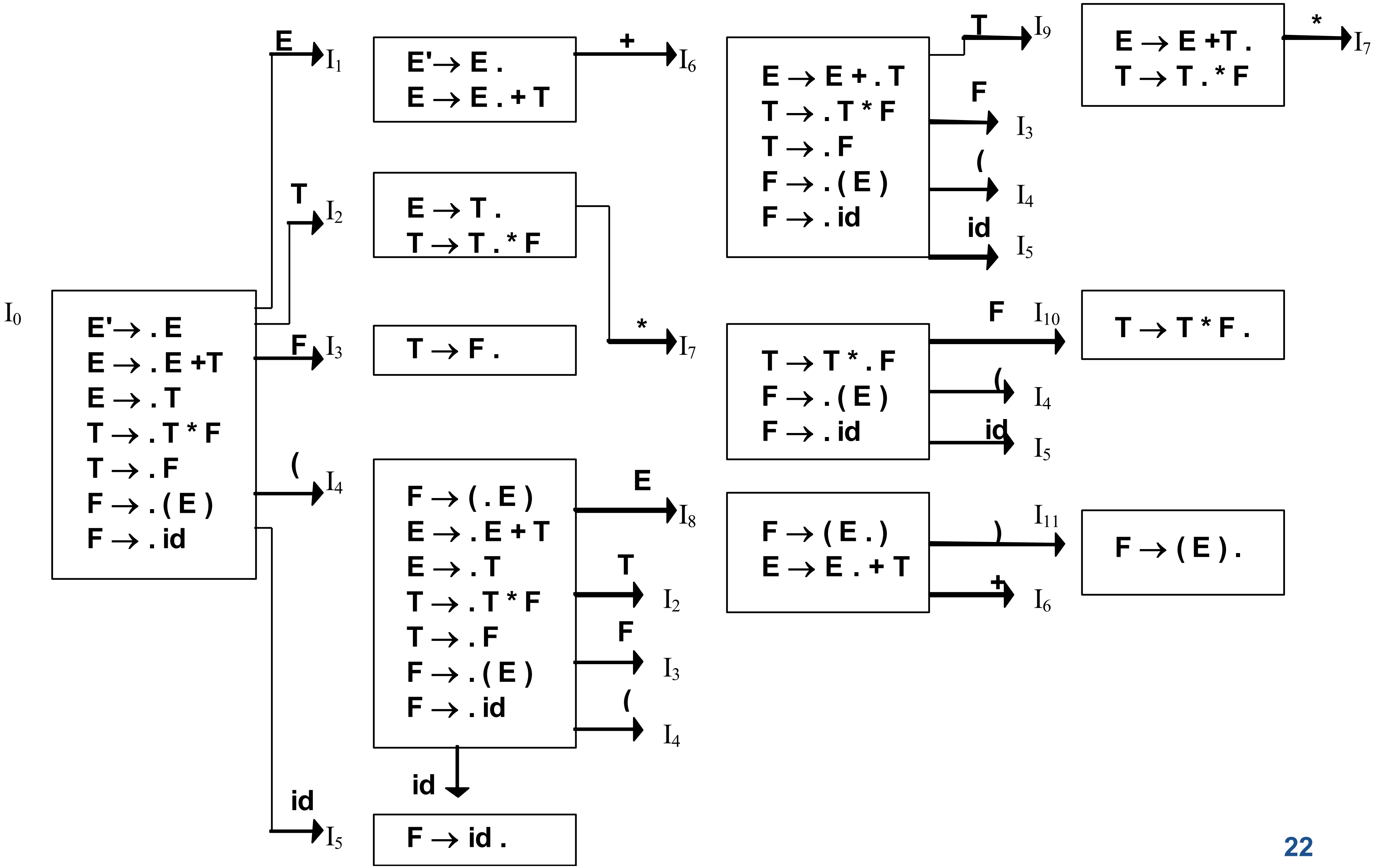
G_2 :

S -> A b (1)

A -> a A (2)

A -> b (3)

Ejemplo: Gramática G_1



Ejemplo: Gramática G_3

S -> A 1
S -> B 2
A -> ε
A -> a A
B -> ε
B -> a B

3. Análisis SLR(1)

Algoritmo construcción de la T.A. SLR(1)

ENTRADA $G' = (N', \Sigma, P', S')$;

SALIDA TA (acc,suc) acc: $Q \times (\Sigma \cup \{\$\}) \rightarrow$ {des-s, red-r, err, acep}

suc: $Q \times N \rightarrow Q \cup$ {err}

METODO

1. $C := \{I_0, I_1, \ldots, I_n\}$; /*Colección canónica de cjtos. de elementos LR(0) */
2. Inicializar TA (acc,suc) con la acción "err";
3. para todo $I_i \in C$

   para todo elemento LR(0) $\in I_i$ hacer

   si $([A \rightarrow \boldsymbol{\alpha} \cdot a\ \beta] \in I_i$: $a \in \Sigma) \wedge$ sucesor $(I_i, a) = I_s$ entonces acc[i,a] := des-s;

   si $[S' \rightarrow S \cdot] \in I_i$ entonces acc[i,\$] := acep;

   si $([A \rightarrow \boldsymbol{\alpha} \cdot] \in I_i) \wedge (r: A \rightarrow \boldsymbol{\alpha}) \in P$ entonces

   paratodo $a \in$ siguientes (A) hacer acc[i,a] := red-r;
4. paratodo $A \in N$ hacer

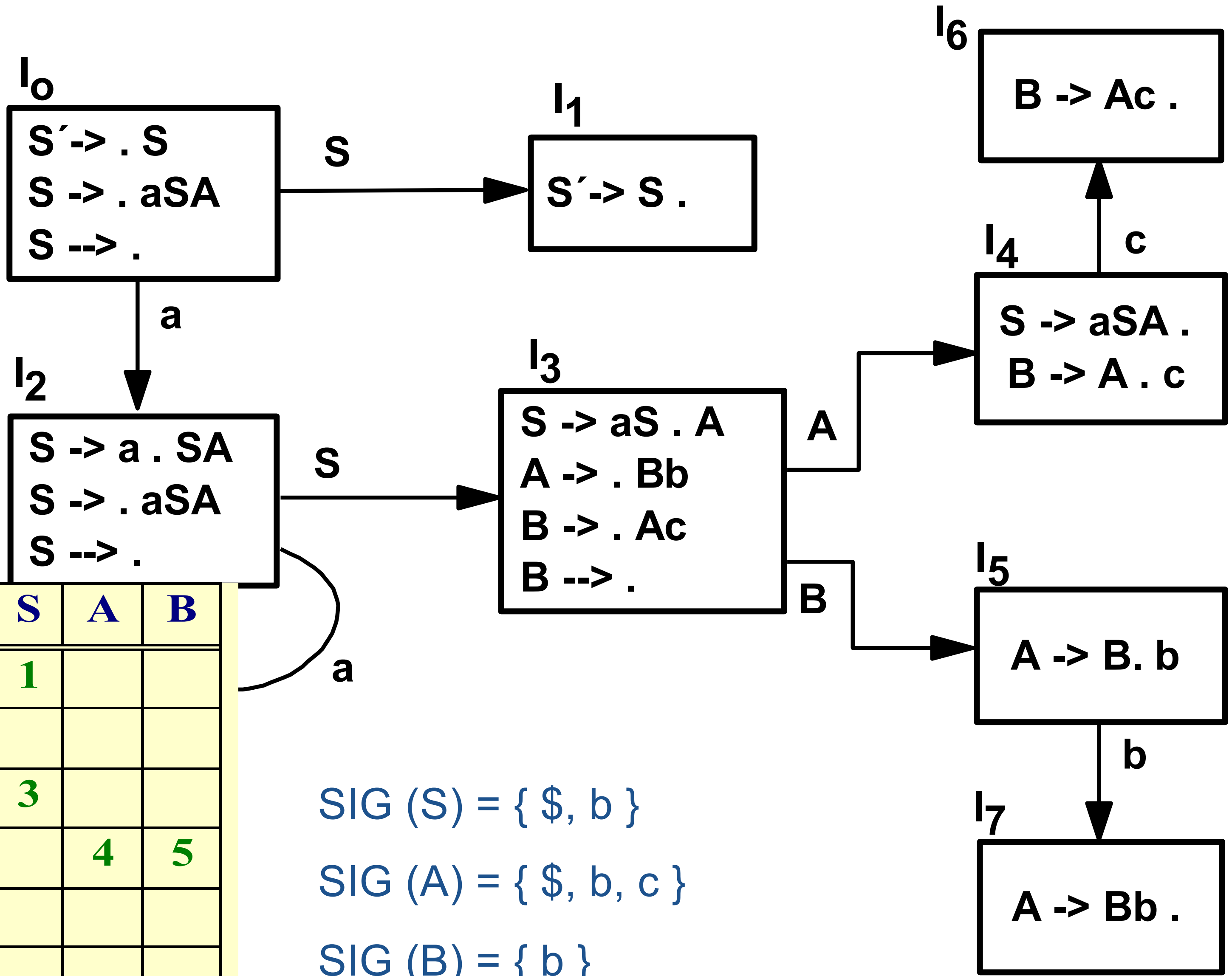
   Si sucesor $(I_i, A) = I_s$ entonces suc [i,A] := s;

FIN

Ejemplo SLR(1) (Gramática G_4)

$S' \rightarrow S$
$S \rightarrow aSA$
$S \rightarrow \varepsilon$
$A \rightarrow Bb$
$B \rightarrow Ac$

$B \rightarrow \varepsilon$



	a	b	c	$	S	A	B
0	d2	r2		r2	1		
1				Acep			
2	d2	r2		r2	3		
3		r5				4	5
4		r1	d6	r1			
5		d7					
6		r4					
7		r3	r3	r3			

SIG (S) = { $, b }

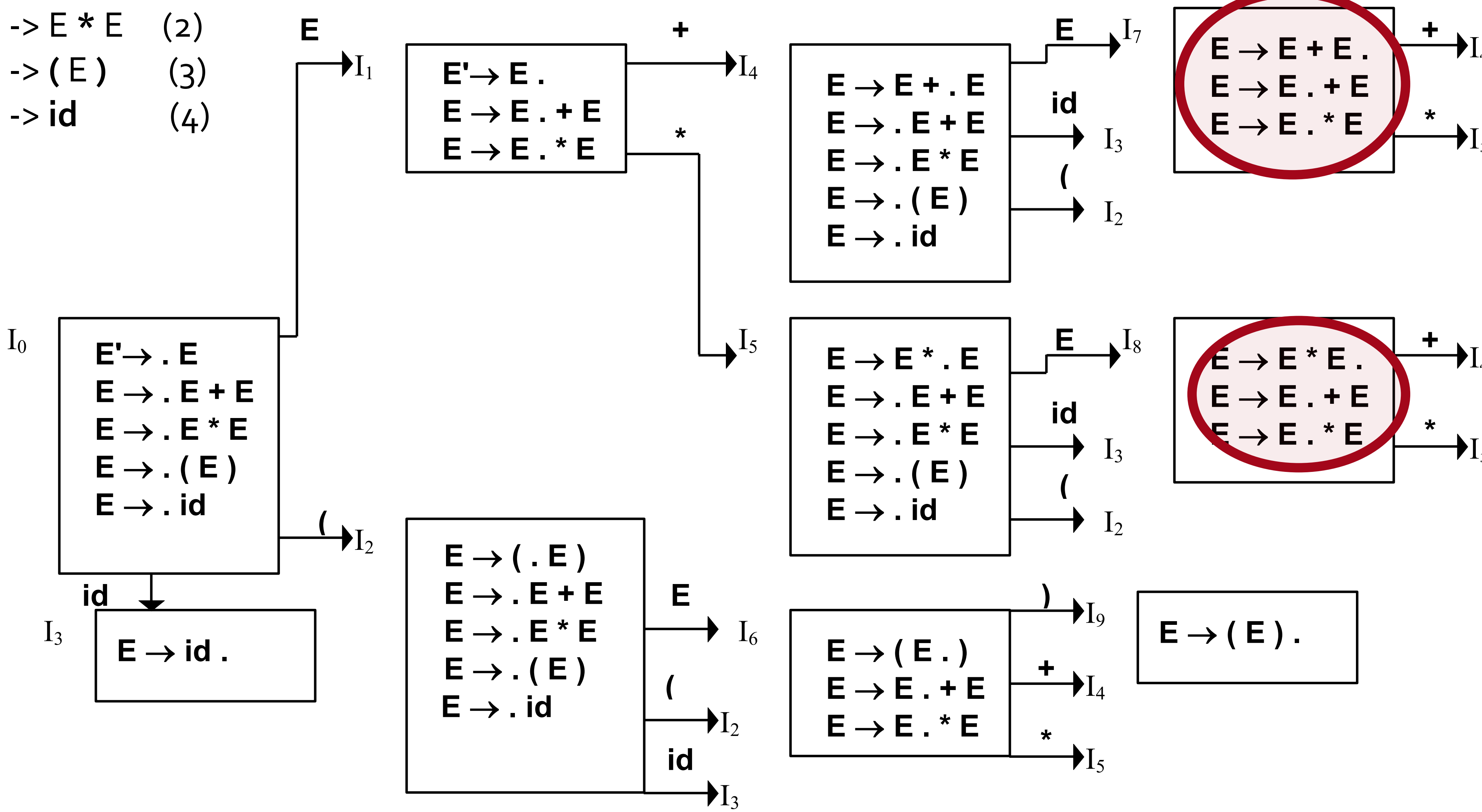
SIG (A) = { $, b, c }

SIG (B) = { b }

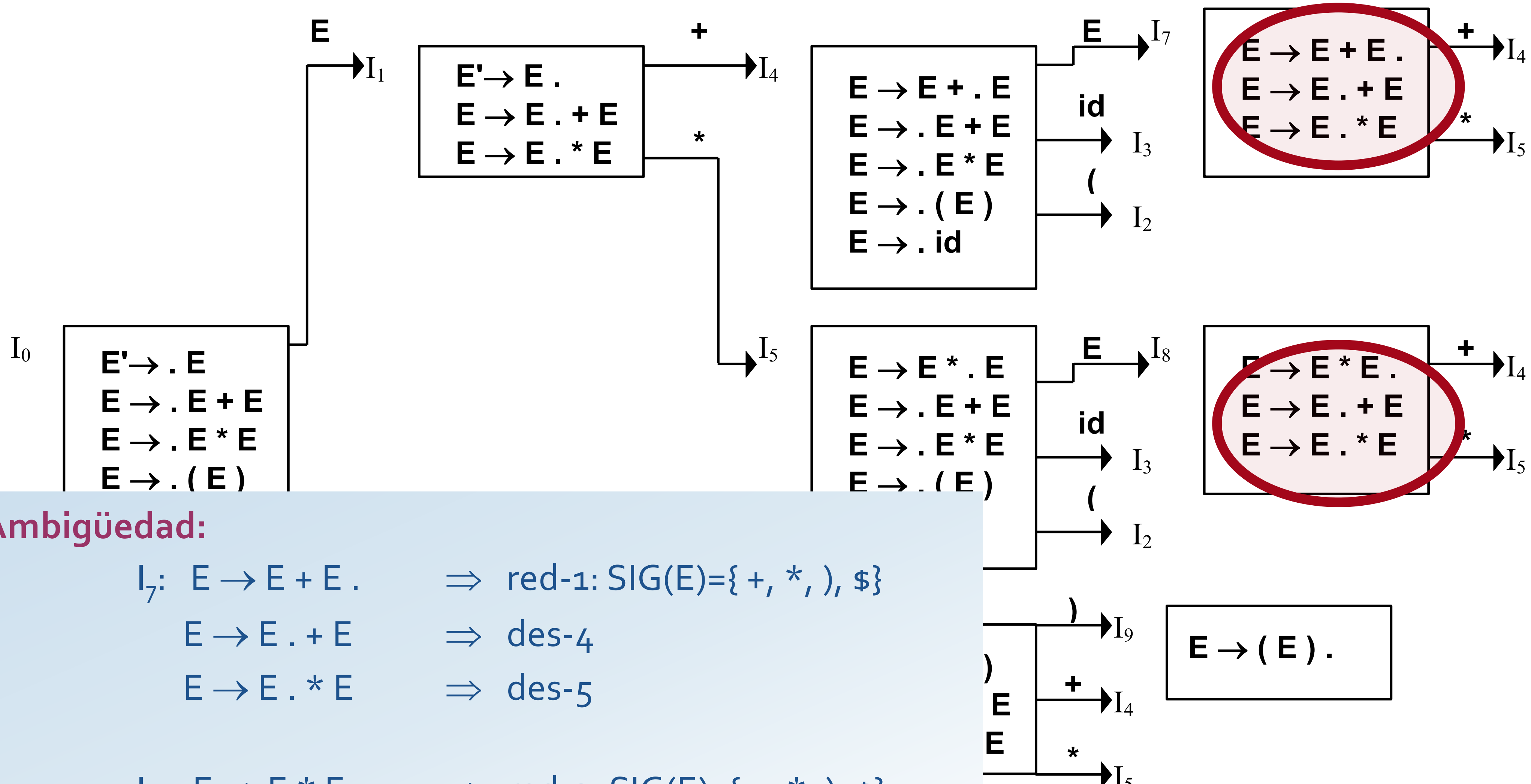
4. Resolución de conflictos

Resolución de conflictos. Gramática G_5

E -> E + E (1)
E -> E * E (2)
E -> (E) (3)
E -> id (4)



Resolución de conflictos. Gramática G_5



Ambigüedad:

I_7: $E \rightarrow E + E .$ $\Rightarrow$ red-1: SIG(E)={ +, *,), $}

$E \rightarrow E . + E$ $\Rightarrow$ des-4

$E \rightarrow E . * E$ $\Rightarrow$ des-5

I_8: $E \rightarrow E * E .$ $\Rightarrow$ red-2: SIG(E)={ +, *,), $}

$E \rightarrow E . + E$ $\Rightarrow$ des-4

$E \rightarrow E . * E$ $\Rightarrow$ des-5

Resolución de conflictos

Criterios para resolver los conflictos:

Mayor precedencia del * frente al +.

Asociatividad a izquierdas de los operadores.

I_7: acción (7,+)=r-1 por asoc. a izq.

acción (7,*)=d-5 por mayor preced. del * que del +

I_8: acción (8,+)=r-2 por mayor preced. del * que del +

acción (8,*)=r-2 por asoc. a izq.

5. Recuperación de errores

Recuperación de errores

Objetivo:

El análisis sintáctico no debe detenerse al encontrar un error. Debe "recuperarse" y continuar analizando en busca de más errores.

Recuperación en modo pánico:

Al detectarse un error sintáctico el analizador saltará símbolos de la cadena de entrada hasta "recuperarse" (sincronizarse) y llegar a un estado en el que pueda continuar el análisis.

Recuperación en modo pánico en el A.S.Descendente:

Si el analizador detecta el error cuando debe aplicar una regla para el no-terminal A, puede "recuperarse" saltando símbolos de la cadena de entrada hasta encontrar un símbolo que pertenezca al conjunto de *símbolos de sincronización* del no-terminal A

El conjunto de símbolos de sincronización para un no-terminal A puede definirse empíricamente o usando un conjunto predeterminado, por ejemplo SIG(A) o PRIM(A).

Recuperación de errores

Recuperación de errores en el A. S. Ascendente:

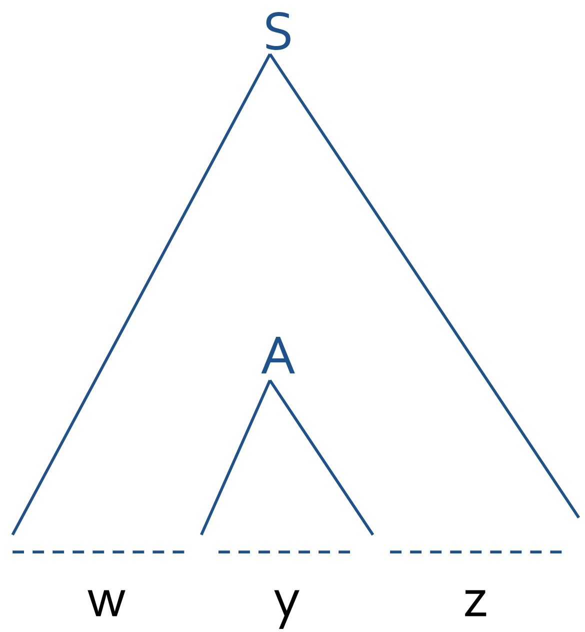
- Ningún símbolo erróneo puede pasar a la pila desde la cadena de entrada.
- El error se detecta siempre con la función **acción** y nunca con la función **sucesor**.

Recuperación en modo pánico en el A. S. Ascendente:

- Las reducciones se realizan en cualquier caso (si en el estado hay reducción).
- Si en el estado que se detecta el error no hay reducciones:
  1. Se eliminan de la pila estados y símbolos, hasta alcanzar un estado "*q*" con una transición con un no terminal (**sucesor**(*q*, A) = r)
  2. Se consumen símbolos de la cadena de entrada hasta llegar a un símbolo "a" que cumple: **acción**(*r*, a) <> error.

6. Relaciones entre gramáticas

Relaciones entre gramáticas



LL(K) → sabemos qué producción aplicar para A conociendo w y los k primeros símbolos de yz ($w \in S^*$)

LR(k) → sabemos qué reducción aplicar sobre wy conociendo wy y los k primeros símbolos de z ($wy \in S^*$)

Jeraquía de gramáticas

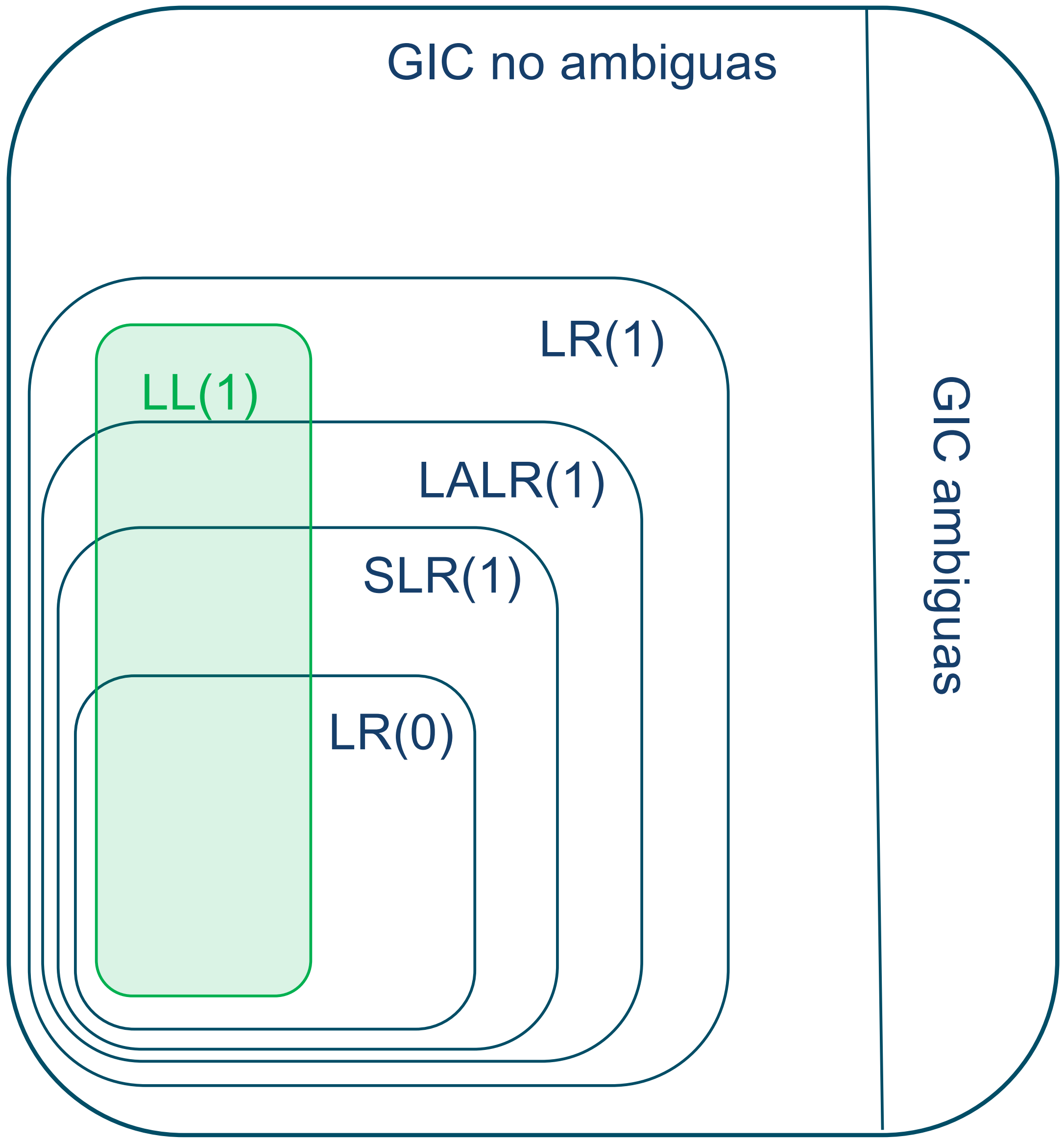
$LR(0) \subset SLR(1) \subset LALR(1) \subset LR(1)$

$\forall\ k \quad LL(k) \subset LR(k)$

$LL(1) \subset LR(1)$

$LL(1) \not\subset SLR(1)$

$LL(1) \not\subset LALR(1)$



Jeraquía de gramáticas. Ejemplos

$G \in SLR(1)$ y $G \notin LL \supset LL(1)$ y $G \notin LR(0)$

$S \rightarrow A1 \quad A \rightarrow a\,A \mid \varepsilon$

$S \rightarrow B2\; B \rightarrow a\,B \mid \varepsilon$

$G \in LL(1)$ y $G \notin SLR(1)$

$S \rightarrow A\,a\,A\,b \mid B\,b\,B\,a$

$A \rightarrow \varepsilon$

$B \rightarrow \varepsilon$

Ejercicio 1

Dada la gramática

S ->	A B		(1)
	|	A C	(2)
B ->	S C		(3)
A ->	**id**		(4)
C ->	**ent**		(5)
	|	**float**	(6)

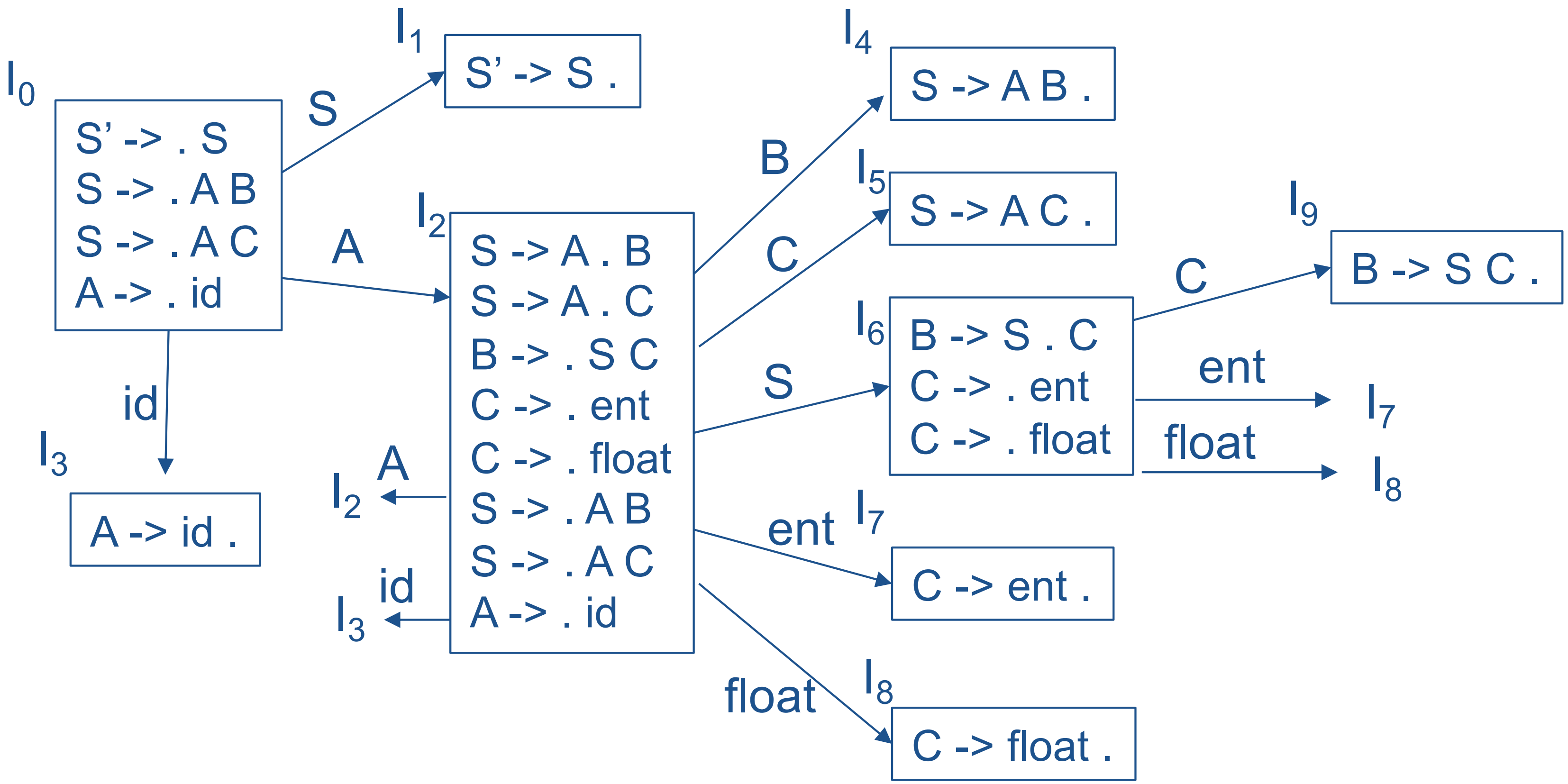
a) Construye la colección canónica de conjuntos de elementos LR(0)

b) Construye la tabla de análisis SLR(1)

c) Realiza la traza SLR(1) para la cadena **ω= id id ent float**

Solución Ejercicio 1

S ->	A B	(1)
|	A C	(2)
B ->	S C	(3)
A ->	**id**	(4)
C ->	**ent**	(5)
|	**float**	(6)



Solución Ejercicio 1

S ->	A B	(1)
|	A C	(2)
B ->	S C	(3)
A ->	**id**	(4)
C ->	**ent**	(5)
|	**float**	(6)

SIG(S) = { $, ent, float }
SIG(B) = { $, ent, float }
SIG(A) = { id, ent, float }
SIG(C) = { $, ent, float }

	id	**ent**	**float**	**$**	**S**	**A**	**B**	**C**
0	d3				1	2		
1				Acept				
2	d3	d-7	d-8		6	2	4	5
3	r-4	r-4	r-4					
4		r-1	r-1	r-1				
5		r-2	r-2	r-2				
6		d-7	d-8					9
7		r-5	r-5	r-5				
8		r-6	r-6	r-6				
9		r-3	r-3	r-3				

(0, id id ent float$,)
(0 id 3, id ent float$,)
(0 id 3, id ent float$,)
(0 A2, id ent float$, 4)
(0 A2 id3, ent float$, 4)
(0 A2 A2, ent float$, 4-4)
(0 A2 A2 ent7, float$, 4-4)
(0 A2 A2 C5, float$, 4-4-7)
(0 A2 S6, float$, 4-4-7-2)
(0 A2 S6 float8, $, 4-4-7-2)
(0 A2 S6 C9, $, 4-4-7-2-6)
(0 A2 B4, $, 4-4-7-2-6-3)
(0 S1, $, 4-4-7-2-6-3-1)
ACEPTADA

Ejercicio 2

Dada la gramática

S -> I = E
I -> id
  | id [E]
E -> cte
  | I
  | ε

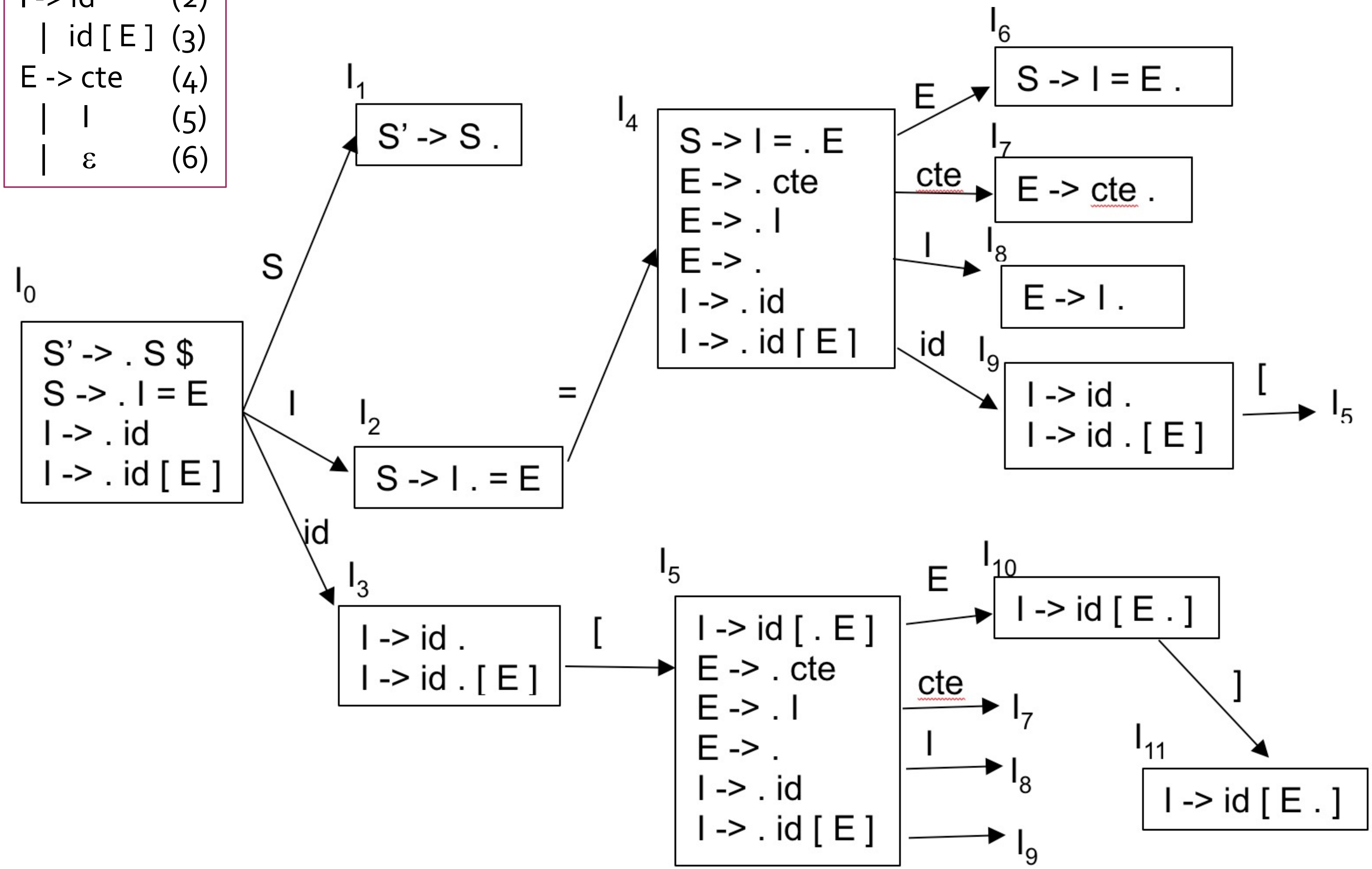
a) Construye la colección canónica de conjuntos de elementos LR(0)

b) Construye la tabla de análisis SLR(1)

c) Realiza la traza SLR(1) para la cadena **ω=** id [ ] = cte

Solución ejercicio 2

S -> I = E (1)
I -> id (2)
| id [E] (3)
E -> cte (4)
| I (5)
| ε (6)



Solución ejercicio 2

S -> I = E (1)
I -> id (2)
| id [E] (3)
E -> cte (4)
| I (5)
| ε (6)

SIG(S) = {$}
SIG(I) = {=, $,] }
SIG(E) = {$,] }

	=	id	[	]	cte	$	S	I	E
0		d-3					1	2	
1						ACEPTAR			
2	d-4								
3	r-2		d-5	r-2		r-2			
4		d-3		r-6	d-7	r-6		8	6
5		d-3		r-6	d-7	r-6		8	9
6						r-1			
7				r-4		r-4			
8				r-5		r-5			
9				d-10					
10	r-3			r-3		r-3			

(0, id[ ]=cte$,) |-
(0id3, [ ]=cte$,) |-
(0id3[5,]=cte$,) |-
(0id3[5E9,]=cte$, 6) |-
(0id3[5E9]10, =cte$, 6) |
(0I2, =cte$, 6-3) |-
(0I2=4, cte$, 6-3) |-
(0I2=4cte7, $, 6-3) |-
(0I2=4E6, $, 6-3-4) |-
(0S1, $, 6-3-4-1) |-
ACEPTAR