Tema 4: Análisis sintáctico ascendente

Procesamiento de Lenguajes

Dept. de Lenguajes y Sistemas Informáticos
Universidad de Alicante

Análisis sintáctico ascendente: un ejemplo

El análisis sintáctico ascendente trata de reconstruir la inversa de una derivación por la derecha de la cadena de entrada
Ejemplo: **int a,b,c;**

$$
\begin{array}{lcl}
D & \rightarrow & T\ \textbf{id}\ L \\
T & \rightarrow & \textbf{int} \\
T & \rightarrow & \textbf{float} \\
L & \rightarrow & \textbf{coma}\ \textbf{id}\ L \\
L & \rightarrow & \textbf{pyc}
\end{array}
$$

$$
\begin{array}{ll}
\textbf{int}\ \textbf{id(a)}\ \textbf{coma}\ \textbf{id(b)}\ \textbf{coma}\ \textbf{id(c)}\ \textbf{pyc} & \Leftarrow \\
T\ \textbf{id(a)}\ \textbf{coma}\ \textbf{id(b)}\ \textbf{coma}\ \textbf{id(c)}\ \textbf{pyc} & \Leftarrow \\
T\ \textbf{id(a)}\ \textbf{coma}\ \textbf{id(b)}\ \textbf{coma}\ \textbf{id(c)}\ L & \Leftarrow \\
T\ \textbf{id(a)}\ \textbf{coma}\ \textbf{id(b)}\ L & \Leftarrow \\
T\ \textbf{id(a)}\ L & \Leftarrow \\
D &
\end{array}
$$

¿Cómo funciona el análisis ascendente?

El analizador, cuando llega un token de la entrada, puede hacer dos cosas:

1. Si el token es una parte derecha de una regla (o completa una parte derecha en construcción), se puede deshacer la regla. Esta acción se denomina reducción

	ACCIÓN
int	reducir $T \rightarrow$ **int**
T	

2. Si no se puede completar una parte derecha, se acumula a lo que se está construyendo. Esta acción se denomina desplazamiento

TOKEN		ACCIÓN
a	T	desplazar a
,	T **id(a)**	desplazar ,
b	T **id(a) coma**	desplazar b
;	T **id(a) coma id(b)**	desplazar ;
	T **id(a) coma id(b) pyc**	reducir $L \rightarrow$ **pyc**
	T **id(a) coma id(b)** L	reducir $L \rightarrow$ **coma id** L
	T **id(a)** L	reducir $D \rightarrow T$ **id** L
	D	

Algoritmo de análisis ascendente por desplazamiento-reducción

- Se utiliza una pila de estados en la que se almacenan las partes derechas en construcción. Cada estado representa un prefijo de una parte derecha, y se asocian a un símbolo terminal o no terminal.
- Consultando unas tablas y el estado en la cima de la pila, el algoritmo puede realizar las siguientes acciones:
  - **Aceptar**: la cadena de entrada es correcta
  - **Desplazar**: se apila un nuevo estado en la pila
  - **Reducir**: se "deshace" una regla, para lo que se desapilan tantos estados como símbolos tenga la parte derecha, y se apila un estado asociado a la parte izquierda
  - **Error**: error sintáctico
- Existen varias formas de construir esas tablas, estudiaremos una de ellas más adelante

Ejemplo de tablas

(1) $D \rightarrow T$ **id** L
(2) $T \rightarrow$ **int**
(3) $T \rightarrow$ **float**
(4) $L \rightarrow$ **coma id** L
(5) $L \rightarrow$ **pyc**

	ACCIÓN						IR A		
	int	**float**	**id**	**coma**	**pyc**	**$**	*D*	*T*	*L*
0	d3	d4					1	2	
1						aceptar			
2			d5						
3			r2						
4			r3						
5				d7	d8				6
6						r1			
7			d9						
8						r5			
9				d7	d8				10
10						r4			

Ejemplo de tablas (2)

	ACCIÓN						IR A		
	int	**float**	**id**	**coma**	**pyc**	**$**	*D*	*T*	*L*
0	d3	d4					1	2	
1						aceptar			
2			d5						
3			r2						
4			r3						
5				d7	d8				6
6						r1			
7			d9						
8						r5			
9				d7	d8				10
10						r4			

PILA	TOKEN	ACCIÓN
0	`int`	d3
0 3	`id(a)`	r2
0 2	`id(a)`	d5
0 2 5	`coma`	d7
0 2 5 7	`id(b)`	d9
0 2 5 7 9	`pyc`	d8
0 2 5 7 9 8	`$`	r5
0 2 5 7 9 10	`$`	r4
0 2 5 6	`$`	r1
0 1	`$`	aceptar

Ejemplo de tablas (3)

	ACCIÓN						IR A		
	int	**float**	**id**	**coma**	**pyc**	**$**	*D*	*T*	*L*
0	d3	d4					1	2	
1						aceptar			
2			d5						
3			r2						
4			r3						
5				d7	d8				6
6						r1			
7			d9						
8						r5			
9				d7	d8				10
10						r4			

PILA	TOKEN	ACCIÓN
0	`int`	d3
0 **int**	`id(a)`	r2
0 *T*	`id(a)`	d5
0 *T* **id**	`coma`	d7
0 *T* **id coma**	`id(b)`	d9
0 *T* **id coma id**	`pyc`	d8
0 *T* **id coma id pyc**	`$`	r5
0 *T* **id coma id** *L*	`$`	r4
0 *T* **id** *L*	`$`	r1
0 *D*	`$`	aceptar

Algoritmo de análisis por desplazamiento-reducción

push(0)
$a :=$ siguienteToken()
REPETIR
    sea s el estado en la cima de la pila
    SI **Accion**$[s, a] =$ **d**j ENTONCES
        push(j)
        $a :=$ siguienteToken()
    SI NO SI **Accion**$[s, a] =$ **r**k ENTONCES
        PARA $i := 1$ HASTA Longitud_Parte_Derecha(k) HACER pop()
        sea p el estado en la cima de la pila
        sea A la parte izquierda de la regla k
        push(**Ir_A**$[p, A]$)
    SI NO SI **Accion**$[s, a] =$ **aceptar** ENTONCES
        fin del analisis
    SI NO
        error()
    FIN_SI
HASTA fin del analisis

Método SLR de construcción de tablas de análisis

Existen varios métodos para construir tablas de análisis, y uno de los más sencillos es el método SLR.
Definiciones:

- Item: una regla de la gramática con un $\bullet$ en alguna posición de la parte derecha, antes y/o después de un símbolo. Ejemplo:

$$
\begin{array}{lcl}
D & \rightarrow & T\ \mathbf{id}\ L \\
\text{Items:} & & \\
D & \rightarrow & \bullet\ T\ \mathbf{id}\ L \\
D & \rightarrow & T\ \bullet\ \mathbf{id}\ L \\
D & \rightarrow & T\ \mathbf{id}\ \bullet\ L \\
D & \rightarrow & T\ \mathbf{id}\ L\ \bullet
\end{array}
$$

IMPORTANTE: Si la regla es $A \rightarrow \epsilon$, el único ítem es: $A \rightarrow \bullet$

Método SLR de construcción de tablas de análisis (2)

Definiciones:

- Clausura de un conjunto de ítems I, $\text{clausura}(I)$:
  1. Todos los ítems de I pertenecen a $\text{clausura}(I)$
  2. Si $A \to \alpha \bullet B\beta$ es un ítem de $\text{clausura}(I)$, se añaden todos los ítems derivados de las reglas de B con el punto al principio: $B \to \bullet\, \alpha_1$, $B \to \bullet\, \alpha_2$, ...
  3. Se repite el paso anterior hasta que no se añaden nuevos ítems a $\text{clausura}(I)$

  Ejemplo:

$$
\begin{array}{rlll}
I = \{ & D & \to & \bullet\ T\ \textbf{id}\ L\ \} \\
\text{clausura}(I) = \{ & D & \to & \bullet\ T\ \textbf{id}\ L\ , \\
 & T & \to & \bullet\ \textbf{int}\ , \\
 & T & \to & \bullet\ \textbf{float}\ \}
\end{array}
$$

Método SLR de construcción de tablas de análisis (3)

Definiciones:

- Función *Ir_A* : dado un conjunto de ítems *I* y un símbolo *A*, la función *Ir_A*(I, A) devuelve un conjunto de ítems que se obtiene, para cada ítem de *I* de la forma $B \to \alpha \bullet A\beta$, añadiendo $\text{clausura}(\{B \to \alpha A \bullet \beta\})$ al conjunto resultado
  Ejemplo:

$$
\begin{aligned}
I = \{ \quad & D \to T \bullet \textbf{id}\ L \ \} \\
Ir\_A(I, \textbf{id}) = \{ \quad & D \to T\ \textbf{id} \bullet L \ , \\
& L \to \bullet\ \textbf{coma}\ \textbf{id}\ L \ , \\
& L \to \bullet\ \textbf{pyc} \ \}
\end{aligned}
$$

Método SLR de construcción de tablas de análisis (4)

Definiciones:

- Colección canónica de conjuntos de ítems, C: dada una gramática, la colección canónica de conjuntos de ítems se construye de la siguiente manera:
  1. Añadir una regla $X \rightarrow S$ a la gramática (temporalmente)
  2. Añadir el conjunto $I_0 = \text{clausura}(\{X \rightarrow \bullet S\})$ a la colección C
  3. Para cada conjunto de ítems I_i de C, y para cada símbolo gramatical A (terminal y no terminal) para el que exista en I_i un elemento del tipo $B \rightarrow \alpha \bullet A\beta$, añadir $Ir\_A(I_i, A)$ a C si no se había añadido antes
  4. Repetir la anterior operación hasta que no se añadan más conjuntos nuevos a la colección de conjuntos de ítems $C = \{I_0, I_1, \ldots\}$

Método SLR de construcción de tablas de análisis (5)

Ejemplo de colección canónica de conjuntos de ítems, C:

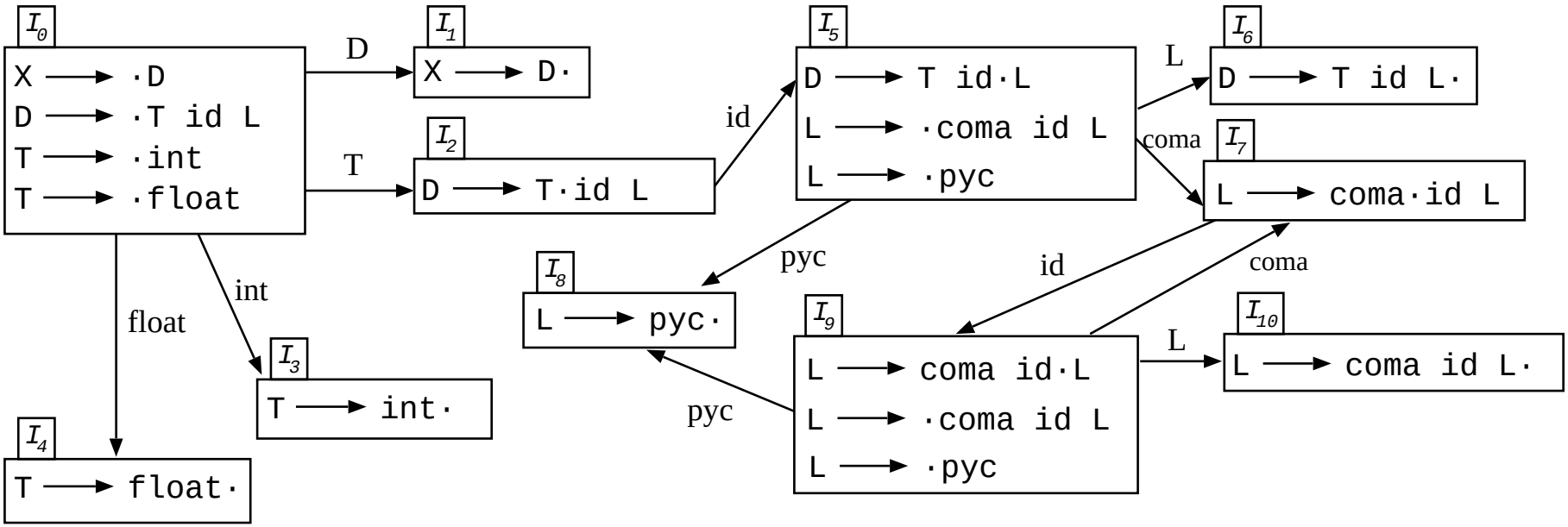
$$
\begin{aligned}
I_0 &= \text{clausura}(\{X \rightarrow \bullet\, D\}) = \{X \rightarrow \bullet\, D\ , \\
&\quad D \rightarrow \bullet\, T\ \mathbf{id}\ L\ , T \rightarrow \bullet\, \mathbf{int}\ , \\
&\quad T \rightarrow \bullet\, \mathbf{float}\ \} \\
I_1 &= Ir\_A(I_0, D) = \{X \rightarrow D\ \bullet\} \\
I_2 &= Ir\_A(I_0, T) = \{D \rightarrow T\ \bullet\ \mathbf{id}\ L\ \} \\
I_3 &= Ir\_A(I_0, \mathbf{int}) = \{T \rightarrow \mathbf{int}\ \bullet\} \\
I_4 &= Ir\_A(I_0, \mathbf{float}) = \{T \rightarrow \mathbf{float}\ \bullet\} \\
I_5 &= Ir\_A(I_2, \mathbf{id}) = \{D \rightarrow T\ \mathbf{id}\ \bullet\ L\ , \\
&\quad L \rightarrow \bullet\, \mathbf{coma}\ \mathbf{id}\ L\ , L \rightarrow \bullet\, \mathbf{pyc}\ \}
\end{aligned}
$$

Método SLR de construcción de tablas de análisis (6)

$$
\begin{aligned}
I_6 &= Ir\_A(I_5, L) = \{D \rightarrow T\ \mathbf{id}\ L\ \bullet\ \} \\
I_7 &= Ir\_A(I_5, \mathbf{coma}) = \{L \rightarrow \mathbf{coma}\ \bullet\ \mathbf{id}\ L\ \} \\
I_8 &= Ir\_A(I_5, \mathbf{pyc}) = \{L \rightarrow \mathbf{pyc}\ \bullet\ \} \\
I_9 &= Ir\_A(I_7, \mathbf{id}) = \{L \rightarrow \mathbf{coma}\ \mathbf{id}\ \bullet\ L\ , \\
&\quad L \rightarrow \bullet\ \mathbf{coma}\ \mathbf{id}\ L\ , L \rightarrow \bullet\ \mathbf{pyc}\ \} \\
I_{10} &= Ir\_A(I_9, L) = \{L \rightarrow \mathbf{coma}\ \mathbf{id}\ L\ \bullet\ \}
\end{aligned}
$$

$$
\begin{aligned}
&Ir\_A(I_9, \mathbf{coma}) = \{L \rightarrow \mathbf{coma}\ \bullet\ \mathbf{id}\ L\ \} = I_7 \\
&Ir\_A(I_9, \mathbf{pyc}) = \{L \rightarrow \mathbf{pyc}\ \bullet\ \} = I_8
\end{aligned}
$$

Método SLR de construcción de tablas de análisis (7)



Autómata reconocedor de prefijos viables

Ejercicio 1

Construye el autómata reconocedor de prefijos viables para la siguiente gramática:

$$
\begin{array}{lcl}
I & \rightarrow & \textbf{print}\ E \\
E & \rightarrow & E\ \textbf{+}\ T \\
E & \rightarrow & T \\
T & \rightarrow & \textbf{(}\ E\ \textbf{)} \\
T & \rightarrow & \textbf{num}
\end{array}
$$

Método SLR de construcción de tablas de análisis (8)

Construcción de las tablas de análisis:

1. Construir el autómata reconocedor de prefijos viables. Cada estado del autómata es un estado del analizador
2. Introducir las transiciones del autómata en las tablas:
   - Si el símbolo de la transición es un terminal, hay que meter un desplazamiento al estado destino en la tabla ACCIÓN
   - Si el símbolo es un no terminal, hay que meter el estado destino en la tabla IR A
3. Si en un estado hay un ítem con el $\bullet$ al final, hay que introducir una reducción por esa regla en ese estado con todos los símbolos que pertenezcan a los SIGUIENTES de la parte izquierda de la regla

Método SLR de construcción de tablas de análisis (9)

Construcción de las tablas de análisis (2):

4. En el estado que tenga el ítem $X \rightarrow S \bullet$, poner la acción "aceptar" con el \$
5. Las casillas vacías de la tabla ACCIÓN son errores sintácticos (los símbolos esperados son las casillas no vacías de ese estado)
6. Si se llega a una casilla vacía en la tabla IR A hay un error en la construcción de las tablas

Conflictos en las tablas SLR

Cuando se construyen las tablas se pueden producir dos tipos de conflictos:

1. **desplazamiento-reducción**: en una casilla en la que hay un desplazamiento también corresponde poner una reducción
2. **reducción-reducción**: en una casilla hay que poner dos reducciones (porque en ese estado hay que reducir por dos reglas diferentes y hay símbolos comunes en los siguientes de las partes izquierdas)

Estos conflictos indican que la gramática no es SLR, por lo que se puede optar por:

1. Utilizar otro método para la construcción de las tablas: LR(1), LALR(1), . . .
2. elegir una de las opciones, p.ej. desplazar en los conflictos desplazamiento-reducción, o reducir por la primera regla en los reducción-reducción, y probar el analizador

Ejercicio 2

Diseña la tabla de análisis SLR para la siguiente gramática:

$$
\begin{array}{lllll}
(1) & I & \longrightarrow & \textbf{if}\ E\ I\ \textbf{fi} \\
(2) & I & \longrightarrow & \textbf{if}\ E\ I\ \textbf{else}\ I\ \textbf{fi} \\
(3) & I & \longrightarrow & \textbf{print}\ E \\
(4) & E & \longrightarrow & \textbf{num}
\end{array}
$$

Una vez hayas diseñado la tabla, realiza la traza del analizador con la cadena:

```
if 11
  if 22
    print 33
  else
    print 44
  fi
fi
```

Ejercicio 3

Diseña la tabla de análisis SLR para la siguiente gramática:

$$
\begin{array}{lllll}
(1) & S & \longrightarrow & L\ \mathbf{,}\ S \\
(2) & S & \longrightarrow & L \\
(3) & L & \longrightarrow & \mathbf{(}\ L\ \mathbf{)} \\
(4) & L & \longrightarrow & \mathbf{a}\ \mathbf{,}\ L \\
(5) & L & \longrightarrow & \epsilon
\end{array}
$$

Una vez hayas diseñado la tabla, realiza la traza del analizador con la cadena:

```
a,(a,),(a,)
```

Ejercicio 4

Diseña la tabla de análisis SLR para la siguiente gramática:

$$
\begin{array}{lllll}
(1) & E & \longrightarrow & E \ \mathbf{+} \ T \\
(2) & E & \longrightarrow & T \\
(3) & T & \longrightarrow & T \ \mathbf{*} \ F \\
(4) & T & \longrightarrow & F \\
(5) & F & \longrightarrow & \mathbf{id} \\
(6) & F & \longrightarrow & \mathbf{(} \ E \ \mathbf{)}
\end{array}
$$

Una vez hayas diseñado la tabla, realiza la traza del analizador con la cadena:

```
id+id*id
```