UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

Escuela Técnica Superior de Ingeniería Informática

Procesadores de Lenguajes

Tema 4

Análisis Sintáctico Ascendente

Javier Vélez Reyes
jvelez@lsi.uned.es

Objetivos del Tema

- Entender la estrategia de reducción-desplazamiento
- Presentar las gramáticas LR
- Entender la arquitectura de un analizador LR
- Entender el algoritmo de análisis LR
- Aprender a construir analizadores LR
- Presentar una clasificación general de gramáticas

Javier Vélez Reyes jvelez@lsi.uned.es

Índice General

- Introducción
- Análisis ascendente reducción - desplazamiento
- Analizador ascendente SLR
- Clasificación de gramáticas

Javier Vélez Reyes jvelez@lsi.uned.es

Introducción

- Análisis sintáctico ascendente
  - Parte de la cadena de entrada para construir la inversa de una derivación por la derecha. Genera el árbol de análisis sintáctico partiendo de las hojas hasta alcanzar el axioma

R1. E := E + E
R2. E := **id**

id + id

r2 r2 r1

id + id <- E + id <- E + E <- E

  - ¿Cuando puede reducirse por una parte izquierda lo que parece ser la parte derecha de una regla?
    - Puede haber partes derechas comunes
    - Puede haber producciones ε

Javier Vélez Reyes jvelez@lsi.uned.es

Índice General

- Introducción
- Análisis ascendente reducción – desplazamiento
  - Tipos de gramáticas
  - Analizadores por reducción – desplazamiento
  - Algoritmo de análisis
- Analizador ascendente SLR
- Clasificación de gramáticas

Javier Vélez Reyes jvelez@lsi.uned.es

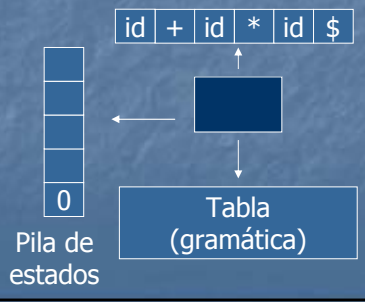
Tipos de gramáticas

- Gramáticas LR
  - Conjunto más amplio de gramáticas que LL(1)
  - Expresión más sencillas
- Tipos de gramáticas
  - LR(1) Analizadores LR(1)
  - LALR Analizadores LALR
  - SLR Analizadores SLR
- Condiciones SLR
  - Se comprueban al construir el analizador

Javier Vélez Reyes jvelez@lsi.uned.es

Análizadores reducción-desplazamiento

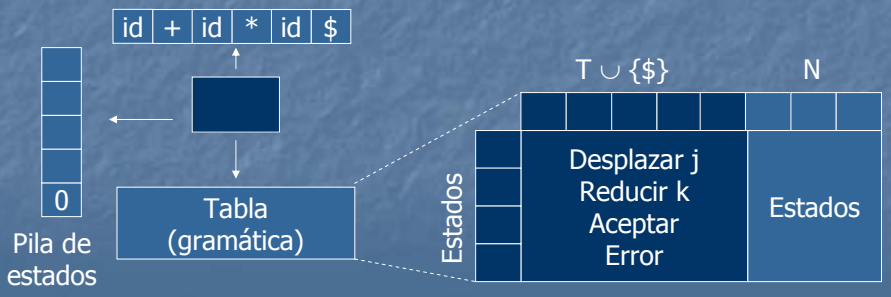
- Análizadores por reducción – desplazamiento
  - Los analizadores son tabulares
  - Se diferencian por el algoritmo de construcción de la tabla
    - Analizador LR
    - Analizador LALR
    - Analizador SLR
  - El algoritmo de análisis es común



Javier Vélez Reyes jvelez@lsi.uned.es

Análizadores reducción-desplazamiento

- Utiliza una tabla para decidir
  - Un estado de pila por fila
  - Una función Acción (s, a) = d_j | r_k
    - dj. Desplazar al siguiente token y apilar el estado j
    - r_k. Reducir por regla k.
  - Una función IrA(si, A) = sj



Javier Vélez Reyes jvelez@lsi.uned.es

Algoritmo de análisis

- Algoritmo de análisis

```
push (0)
a := token();
REPEAT
        Sea s el estado en el tope de la pila
        IF Acción [s, a] = dj
                push (j)
                a := token ();
        ELSEIF Acción [s,a] = rk
                FOR i := 1 TO Longitud (Parte derecha de k)
                        pop ();
                p = cima ();
                Sea A parte izquierda de k
                push (irA [p, A])
        ELSEIF Acción [s,a] = aceptar
                Terminar con éxito
        ELSE error
UNTIL true
```

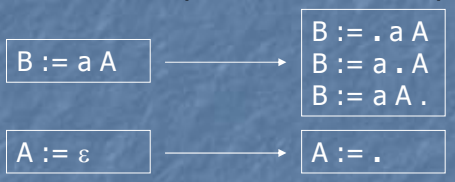
Javier Vélez Reyes jvelez@lsi.uned.es

Índice General

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Términos comunes
  - Elemento. Una regla de producción que incorpora en alguna posición de su parte derecha un punto



  - Prefijo viable. La parte situada a la izquierda del punto de algún elemento

Prefijo viable

S := B A **. end**

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Funciones comunes
  - Clausura (I)

```
Clausura (I) = I

SI      A := α . Bβ ϵ Clausura (I) ^
        B ϵ N ^
        B := δ1 | δ2 | … | δn
ENTONCES
        Clausura (I) = Clausura (I) ∪ {B := . δi} i = 1..n
        Repetir esta regla hasta que no se pueda aumentar I
```

  - IrA (s, A)

```
FOR todos los elementos B := α . Aβ ϵ I DO
        IrA (s, A) = IrA (s, A) ∪  Clausura({B := α A.β })
```

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Colección Canónica del conjunto de elementos
  - Colección de todos los conjuntos de elementos
  - C = { I_0, I_1, ...}
- Construcción

Ampliar la gramática con S' := S

C = Clausura ({S' := .S}) = I_0

Para cada conjunto Ii ϵ C y para cada A ϵ {T ∪ N} para el que exista en Ii un elemento de tipo B := α . Aβ

C = C ∪ IrA (Ii, A)

Repetir el paso anterior hasta que C no pueda ampliarse.

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Autómatas reconocedores de prefijos viables
  - Los estados Sj del autómata son los conjuntos Ij de C
  - Las transiciones son los símbolos N ∪ T ∪ {$}
- Construcción

S0 = Clausura ({S' := .S})

REPEAT

FOR cada A ϵ N ∪ T / Existe B := α . Aβ ϵ Si DO

Crear un estado nuevo Sn = IrA (Si, A) (si no existe)

Crear una transición de Si a Sn etiquetada con A

En cada iteración considerar como Si cada Sn nuevo

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Construcción de tablas SLR

Obtener C = {I0, I1, …, In}

Cada Ij ϵ C se corresponde con el estado j del analizador

Construir la tabla IrA [s, A]
Si IrA (Ii, A) = Ij Entonces IrA [i, A] = j

Construir la tabla Acción [s, A]
Para todo A := α . aβ ϵ Ii existe Ij / irA (Ii, a) = Ij hacer
Acción [i,a] = dj
Para todo A := β. o A := . ϵ Ii hacer
Calcular SIG (A)
Para todo a ϵ SIG (A) hacer
Acción [i, a] = rk (k = número de regla)

Si S' := S. Є Ii Acción [i, $] = Aceptar

Todas las demás entradas de Acción [k, s] = error

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Construcción de tablas SLR a partir del autómata

Las transiciones de Si a Sj etiquetas con A hacen IrA [i, A] = j

Las transiciones de Si a Sj etiquetas con a hacen Acción [i, a] = j

Las reducciones se calculan como antes solo que se usan los estados del autómata

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Conflictos en las tablas SLR
  - Existen múltiples entradas para la entrada acción
- Causas
  - La gramática No es SLR (LRA, LR(1) u otra)
  - La gramática es ambigua
- Tipos
  - Conflicto reducción – desplazamiento
  - Forzar una elección

S := **if** S
S := **if** S **else** S
S := other

Ij = {S := **if** S**.** , S := **if** S **. else** S}
r1 dk

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

- Conflictos en las tablas SLR
  - Existen múltiples entradas para la entrada acción
- Causas
  - La gramática No es SLR (LRA, LR(1) u otra)
  - La gramática es ambigua
- Tipos
  - Conflicto reducción - reducción

S := **id** A | **id** B **fin**
B := **print** A **fin** | ε
A := other | ε

r3 r2
Ij = {A := . , B := .}
SIG (A) Λ SIG (B) ≠Φ

Javier Vélez Reyes jvelez@lsi.uned.es

Analizador sintáctico SLR

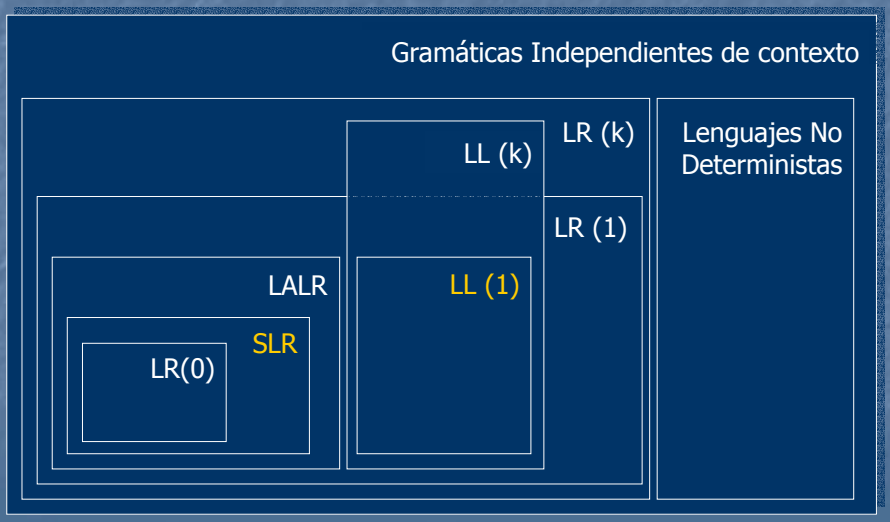
- Errores en analizadores SLR
  - Cada celda vacía de la tabla de acción es un error
- Reducción de mensajes de error
  - En las casillas de error de los estados que reduzca siempre por la misma regla forzar la reducción. El error se propaga hasta el siguiente desplazamiento.
  - Expresar un mensaje de error por cada estado a partir de las acciones correctas de dicho estado.
- Recuperación
  - Eliminar el menor número posible de la entrada antes de reanudar el análisis
  - Pasar a un estado consistente

Javier Vélez Reyes jvelez@lsi.uned.es

Índice General

Javier Vélez Reyes jvelez@lsi.uned.es

Clasificación de gramáticas



Javier Vélez Reyes jvelez@lsi.uned.es

Bibliografía

[AJO] AHO, SETHI, ULLMAN: *Compiladores: Principios, técnicas y herramientas*,: Addison-Wesley Iberoamericana, 1990

[GARRIDO] A. Garrido, J. Iñesta, F. Moreno y J. Pérez. 2002. *Diseño de compiladores*. Universidad de Alicante.