INTRODUCCIÓN A LOS LENGUAJES

1 Definición de lenguaje

2 Lenguajes formales

- 2.1 Requisitos
- 2.2 Aplicaciones
- 2.3 Especificación
- 2.4 Gramáticas
  - 2.4.1 Definición y Ejemplos
  - 2.4.2 Ambigüedad
  - 2.4.3 Clasificación de Chomsky
- 2.5 Lenguajes y teoría de autómatas

INTRODUCCIÓN A LOS LENGUAJES

3 Traductores e intérpretes

- 3.1 Análisis léxico
- 3.2 Análisis sintáctico
- 3.3 Análisis semántico
- 3.4 Generación de la salida

1. DEFINICIÓN DE LENGUAJE

- Medio de expresión y comunicación

- Ejemplos
  - Lenguaje habitual (oral o escrito)
  - Lenguaje de las señales de tráfico
  - Lenguaje de los sordomudos
  - Lenguaje para que los ciegos puedan leer
  - Lenguaje matemático
  - Lenguaje máquina
  - Lenguajes de programación
  - Etc…

1. DEFINICIÓN DE LENGUAJE

- Fundamento básico:

asociación SÍMBOLO-SIGNIFICADO

ALFABETO: conjunto finito de símbolos

- con significado (chino)
- sin significado (castellano, inglés)

LENGUAJE: secuencias de símbolos del alfabeto

1. DEFINICIÓN DE LENGUAJE

B.L. Whorf

Language **shapes** the way we think, and **determines** what we can think about

1 Tenemos que codificar (shapes)

2 Limita lo que podemos expresar (determines)
(imaginemos un lenguaje sin recursividad)

2. LENGUAJES FORMALES

2.1 Requisitos

No ambigüedad

Cada cadena del lenguaje debe tener una única interpretación

Completitud en su definición

Ejemplo: imaginemos que no se define la situación de una variable índice de bucle al finalizar el mismo

2.2 APLICACIONES

- Diseño de hardware (VHDL)
- Diseño de software:
  - Editores con ayuda sintáctica
  - Lenguajes de programación (C, ADA, JAVA, LISP, PROLOG, ..)
  - Lenguajes ensambladores y máquina
  - Lenguajes para procesado de señales (MATLAB)
  - Lenguajes para páginas web (HTML, XML)
  - Lenguajes para bases de datos (SQL)

2.3 ESPECIFICACIÓN

SINTAXIS y SEMÁNTICA

- SINTAXIS: determina las cadenas que pertenecen al lenguaje

- SEMÁNTICA: determina el significado de cada cadena del lenguaje. Se especifica de forma informal en los manuales de referencia de los lenguajes

2.3 ESPECIFICACIÓN

- Sintaxis: especificación
  - Por extensión (sólo para lenguajes finitos)

$$L = \{ aa, bb \}$$

  - Por propiedades

$$L = \{ a^n b^n \mid n>0 \}$$

  - Utilizando un Metalenguaje: las Gramáticas

2.4 GRAMÁTICAS

$$G (Vt , Vn , P , S)$$

Vt Vocabulario de símbolos terminales

Vn Vocabulario de símbolos no terminales

$V = Vt \cup Vn$

P Conjunto de reglas de producción (α , β)

$\alpha \in V^+$ $\beta \in V^*$

Nomenclatura habitual: $\alpha \rightarrow \beta$ $\alpha := \beta$

S Símbolo principal ($S \in Vn$)

EJEMPLOS

$L=\{0^n1^n \mid n\geq 0\}$

G: R1 $S \rightarrow 0S1$

R2 $S \rightarrow \varepsilon$

$S \rightarrow 0S1 \rightarrow 00S11 \rightarrow 000S111 \rightarrow 000111$

Derivación

Forma sentencial

Sentencia

GRAMÁTICAS EQUIVALENTES

G y G' son equivalentes ↔ L(G)=L(G')

Ejemplo: $L=\{a^m b^n \mid m,n\geq 0\}$

G:			G':		
	R1	S → A B		R1	S → A S B
	R2	A → a A		R2	S → ε
	R3	A → ε		R3	A → a
	R4	B → b B		R4	A → ε
	R5	B → ε		R5	B → b
				R6	B → ε

2.4.2 GRAMÁTICAS AMBIGUAS

Una gramática es ambigua si, y solo si, para alguna cadena del lenguaje existe más de un árbol sintáctico.

Ejemplo: la gramática G' anterior es ambigua

1. S → ASB → aSB → aεB → aεb
2. S → ASB → AASBB → AεSBB → aεSBB → aεεBB → aεεbB → aεεbε

Tenemos una cadena con, al menos, 2 árboles sintácticos

2.4.3 CLASIFICACIÓN DE CHOMSKY

Gramáticas Tipo 0

$\alpha \rightarrow \beta \qquad \alpha \in V^{+} \qquad \beta \in V^{*}$

Gramáticas Tipo 1

$\alpha \rightarrow \beta \qquad \alpha \in V^{+} \qquad \beta \in V^{*} \qquad |\beta| \geq |\alpha|$

2.4.3 CLASIFICACIÓN DE CHOMSKY

Gramáticas Tipo 2 (libres de contexto)

$\alpha \rightarrow \beta$ $\quad \alpha \in Vn \quad |\alpha|=1 \quad \beta \in V^*$

Gramáticas Tipo 3 (regulares)

$\alpha \rightarrow \beta$ $\quad \alpha \in Vn \quad |\alpha|=1$

β es de la forma: $a\ N \mid N\ a \mid \varepsilon$

$a \in Vt$

$N \in Vn$

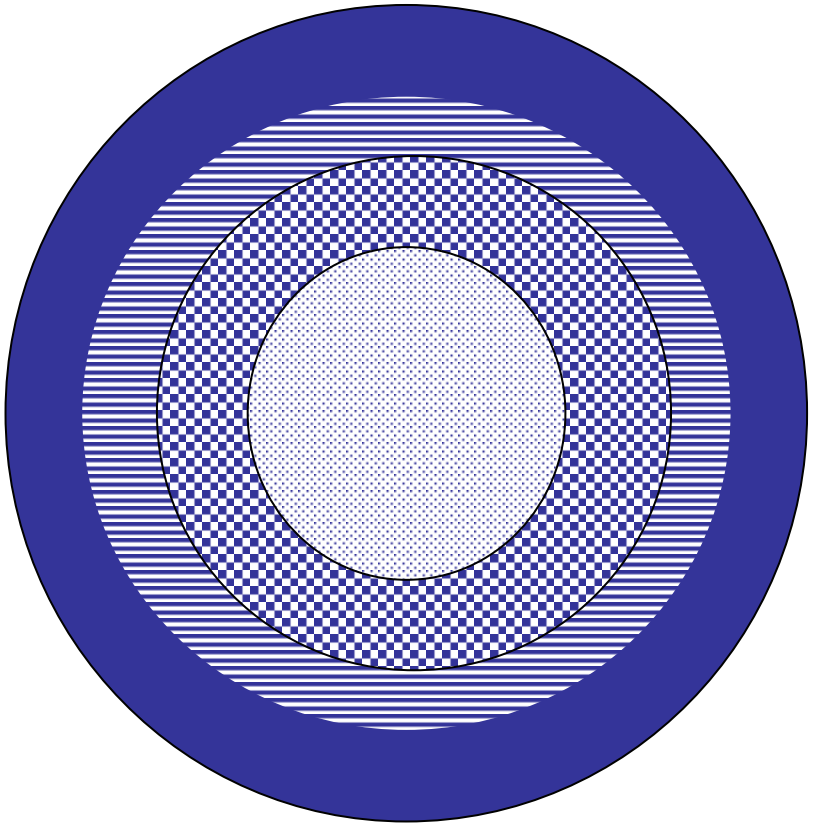
Relación de inclusión

Tipo 0

Tipo 1

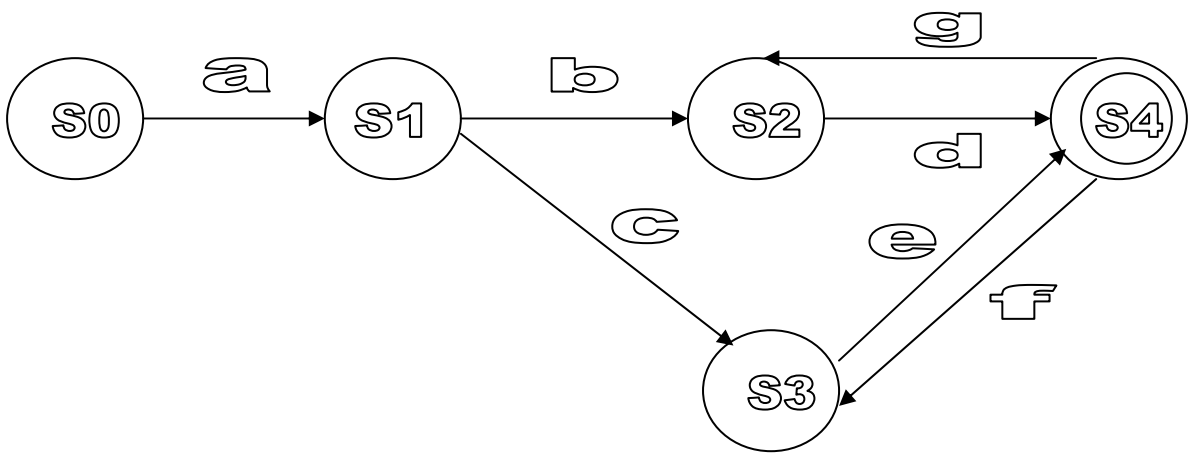
Tipo 2

Tipo 3



2.5 LENGUAJES Y TEORÍA DE AUTÓMATAS

- Los lenguajes regulares son equivalentes a AFD (Autómatas Finito Deterministas)



LENGUAJE REGULAR EQUIVALENTE

S0 → a S1

S1 → b S2 | c S3

S2 → d S4

S3 → e S4

S4 → g S2 | f S3 | ε

3 TRADUCTORES E INTÉRPRETES

3.1 Análisis léxico

Detección de los símbolos terminales del lenguaje

A = B + C * 3

<id> <op_asig> <id> <op_suma> <id> <op_mul> <cte_ent>

Herramienta: LEX

3.2 ANÁLISIS SINTÁCTICO

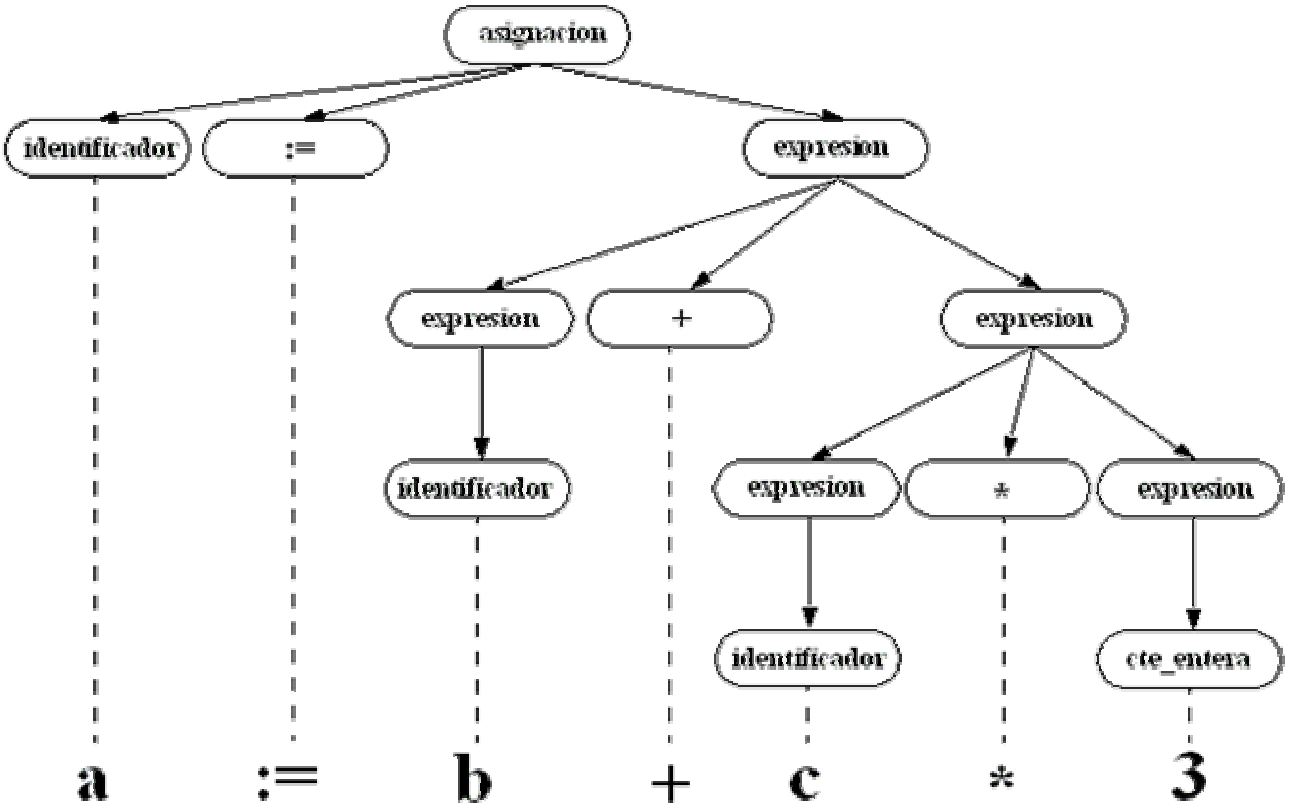
Comprueba la estructura sintáctica de una cadena

asignacion → identificador ':=' expresion

expresion → expresion "+" expresion
| expresion "*" expresion
| "(" expresion ")"
| identificador
| cte_entera

EJEMPLO: a := b + c * 3

3.2 ANÁLISIS SINTÁCTICO



Análisis léxico / Análisis sintáctico

Se trata de una separación un tanto artificial para ganar velocidad en el proceso completo de reconocimiento.

Tal como explicamos anteriormente, los patrones regulares de los elementos léxicos pueden expresarse mediante gramáticas regulares.

En ese caso, el análisis léxico desaparecería casi por completo, quedando reducido al reconocimiento de caracteres.

3.3 ANÁLISIS SEMÁNTICO

Una cadena de un lenguaje puede ser sintácticamente correcta, y sin embargo puede ser incorrecta desde el punto de vista del significado (tipos, parámetros, etc.)

```
s=suma (a,b,c);
…
suma (int x, int y)
{
        return(x+y);
}
```

3.4 GENERACIÓN DE LA SALIDA

1. Intérpretes

   (ejecutan aquello especificado en el lenguaje)

   Ejemplo: a := b + c * 3 (evalúa y asigna)

2. Traductores

   (generan una especificación de salida, que luego será traducida o interpretada)

   Ejemplo: a := b + c *3 (genera código)

3.4 GENERACIÓN DE LA SALIDA

a) Código intermedio

```
temp1= c * 3;
temp2= b + temp1;
a=temp2;
```

b) Código ensamblador

```
MOV   c, R2
MUL   #3, R2
MOV   b, R1
ADD   R2, R1, R3
MOV   R3, a
```