Tema1.pdf

PruebaAlien

Modelos de Computación

3º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Hay 2 tipos de problemas con la gramática:

- Tipo 1: Lenguajes regulares
- Tipo 2: Lenguajes Libres del contexto

Forma automática: Usaremos algoritmos

Forma eficiente: Tiempo real (Veremos AFD [Autómatas Finitos] y después Autómatas con pila)

Algoritmo Turing: Es un paso previo al problema, esto sirve para demostrar que ese programa no existe.

L	If Stops(P,P) GOTO L

Este es un programa turing(P)

GOTO L → Vuelve a ejecutar el programa

De tal forma que el programa no se detiene, si stop devuelve **true**, haciendo un ciclo infinito.

Se hace 2 casos:

- **Caso 1:** Stops(Turing, Turing) = true (Si se detiene el programa de Turing el método Stops devolverá TRUE)
- **Caso 2:** Stops(Turing, Turing) = false (El programa acaba y finaliza)

Significados

Alfabeto: es un conjunto finito A y sus elementos se llaman **símbolos** o **letras**.

Ejemplo:
Tenemos 2 sensores que devuelven 0 o 1:
- alfa: 0, 1
- beta: 0, 1

0 no hay coches y 1 hay coches

(0,0) --> no hay coches ni en alfa ni beta
(0,1) --> hay coches en alfa
(1,0) --> hay coches en beta
(1,1) --> hay coches en alfa y beta

con lo cual un alfabeto son las posibles combinaciones o casos que haya

A = {0,1}

B = {< 0,0 >,< 0,1 >,< 1,0 >,< 1,1 >}

Palabra: Sobre el alfabeto A es una sucesión finita de elementos de A.

Palabra es una secuencia de simbolos:
(0,0), (1,0), (1,0), (0,0), (1,1)...

Si A = {0,1} entonces 0111 es una palabra sobre este alfabeto.

El **conjunto de todas las palabras** sobre un alfabeto **A** se nota como **A*.**

La palabra vacía es la palabra de longitud 0 Notación: ε

Para decir que el conjunto de cadenas sobre un alfabeto **A** excluyendo la cadena vacía se nota como A^+.

Si $u, v \in A^*, u = a_1 \dots a_n, v = b_1 \dots b_m$, se llama **concatenación**

de u y v a la cadena $u \cdot v$ (o simplemente uv) dada por $a_1 \dots a_n b_1 \dots b_m$

Ejemplo:

Si **u = 011**, **v = 1010**, entonces **uv = 0111010**

Propiedades:

1. $|u \cdot v| = |u| + |v|, \ \forall u, v \in A^*$
2. **Asociativa**: $u \cdot (v \cdot w) = (u \cdot v) \cdot w, \ \forall u, v, w \in A^*$
3. **Elemento Neutro:** $u \cdot e = e \cdot u = u, \ \forall u \in A^*$

Iteración n-ésima de una cadena (u^n) como la concatenación con ella misma **n** veces.

Si $u \in A^*$ entonces

- $u^0 = \varepsilon$
- $u^{i+1} = u^i \cdot u, \forall i \geq 0$

Ejemplo

Si **u = 010**, entonces u^3 **= 010 010 010**.

Si $u = a_1 \dots a_n \in A^*$, entonces la **cadena inversa** de **u** es la cadena $u^{-1} = a_n \dots a_1 \in A^*$.

Ejemplo

Si **u = 011**, entonces u^{-1} **= 110**.

Un **lenguaje** sobre un alfabeto **A** es un subconjunto del conjunto de las cadenas sobre **A**: $L \subseteq A^*$

Es decir, un lenguaje es una secuencia de símbolos.

Notación: **Lenguajes**: L, M, N, …

Ejemplo:

- $L_1 = \{a, b, e\}$ **(TRES PALABRAS)**
- $L_2 = \{a^i b^i \mid i = 0,1,2, \dots\}$ **(El alfabeto es A = {a,b}; u = aabb = a^2 b^2)**
- $L_3 = \{uu^{-1} \mid u \in A^*\}$ **(Palíndromos de longitud par)**
- $L_4 = \{a^{n^2} \mid n = 1,2,3, \dots\}$ (Sucesiones de **a** de longitud un cuadrado perfecto)

KEEP CALM AND ESTUDIA UN POQUITO

WUOLAH + #QuédateEnCasa

L_2 es una sucesión de a seguida de una b de la misma longitud.

L_3: (abba) **palíndromo** es que leas como leas la cadena es la misma.

A = {a,b}

A* = son todas las cadenas posibles con el alfabeto A

(Ejemplo: **u = aabab**) $u^{-1} = babaa$ (es la cadena inversa)

$uu^{-1} = aababbabaa$

Conjuntos numerables: Aquel conjunto al que se le puede asignar un numero a cada elemento del conjunto, de tal forma que 2 elementos distintos tengan distinto número. En resumen, cada elemento está asociado a un número distinto.

Conjuntos no numerables: Pueden estar asociados a un mismo numero varios elementos distintos del conjunto.

Si L_1, L_2 son 2 lenguajes sobre el alfabeto A, la **concatenación** de estos 2 lenguajes se define como: $L_1L_2 = \{u_1u_2 \mid u_1 \in L_1, u_2 \in L_2\}$

Propiedades:

- $L0 = 0L = 0$
- Elemento Neutro: $\{\varepsilon\}L = L\{\varepsilon\} = L$
- Asociativa: $L_1(L_2L_3) = (L_1L_2)L_3$

Si $L_1 = \{0^i1^i : i \geq 0\}$; $L_2 = \{1^i0^i : i \geq 0\}$, entonces,

$L_1L_2 = \{0^i1^i1^j0^j : i, j \geq 0\}$

Ejemplo:

u1 = 0011, 000111 u2 = 10, 1100 u1u2 = 0011 10 u1 u2	Si j = 0, i = 2 $0^2\ 1^2\ 1^0\ 0^0 = 0^2\ 1^2 = u1$

La **iteración** de lenguajes se define de forma recursiva: $L^0 = \{\varepsilon\}$, $L^{i+1} = L^iL$

Si L es un lenguajes sobre el alfabeto A, la clausura de Kleene de L es: $L^* = \cup_{i\geq 0} L^i$

$L^+ = \cup_{i\geq 1} L^i$

Propiedades:

- $L^+ = L^*$ si $\varepsilon \in L$
- $L^+ = L^* - \{\varepsilon\}$ si $\varepsilon \notin L$

Ejemplo: (cualquier cosa que quieras concatenar con 0 o 01 pertenecerá a L* y a L+)

Si $L = \{0, 01\}$, entonces:

L^* = **Conjunto de palabras sobre** $\{0,1\}$ ***en las que un 1 va siempre precedido de un cero.***

L^+ = **Conjunto de palabras sobre** $\{0,1\}$ ***en las que un 1 va siempre precedido de un cero y distintas de la palabra vacia***

Lenguaje inverso de L es: $L^{-1} = \{u \mid u^{-1} \in L\}$

Cabecera: $CAB(L) = \{u \mid u \in A^* \text{ y } \exists v \in A^* \text{ tal que } uv \in L\}$

Ejemplo: Si $L = \{0^i 1^i : i \geq 0\}$ entonces $CAB(L) = \{0^i 1^j : i \geq j \geq 0\}$

Si A_1 y A_2 son 2 alfabetos, una aplicación: $h: A_1^* \rightarrow A_2^*$

Se dice que es un **homomorfismo** si y solo si $h(uv) = h(u)h(v)$

En resumen, es como una traducción.

Consecuencias:

- $h(\varepsilon) = \varepsilon$
- $h(a_1 \dots a_n) = h(a_1) \dots h(a_n)$

Ejemplo:

Si $A_1 = \{0,1,2,3,4,5,6,7,8,9\}, A_2 = \{0,1\}$

$$h(0) = 0000, \quad h(1) = 0001, \quad h(2) = 0010, \quad h(3) = 0011$$
$$h(4) = 0100, \quad h(5) = 0101, \quad h(6) = 0110, \quad h(7) = 0111$$
$$h(8) = 1000, \quad h(9) = 1001$$
$$h(034) = 0000\ 0011\ 0100, \quad h(\varepsilon) = \varepsilon$$

Una **gramática generativa** es un cuadrupla (V, T, P, S) en la que:

- **V** es un alfabeto, llamado **variables** o símbolos no terminales, es decir son los utensilios o recipientes a usar (se representa con letras mayúsculas)
- **T** es un alfabeto, llamado **Simbolos terminales,** es decir son los ingredientes o componentes (se representa con letras minúsculas)
- **P** es un conjunto de pares (α, β), llamados **reglas de producción**, donde $\alpha, \beta \in (V \cup T)^*$ y α contiene al menos un símbolo de **V.** El par (α, β) suele representarse como $\alpha \rightarrow \beta$**.** Es decir, es una secuencia de pasos a seguir.
- **S** es un elemento de **V,** llamado **símbolo de partida.**

Ejemplo:

$\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{Y}, \boldsymbol{P}, \boldsymbol{S})$ dada por,

$\boldsymbol{V} = \{\boldsymbol{E}\}$ (conjunto de variables)

$\boldsymbol{T} = \{+, *, (,), \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}$ (los elementos para elaborar)

$\boldsymbol{P}$ está compuesto por las siguientes **reglas de producción**

$$\boldsymbol{E \to E + E}, \quad \boldsymbol{E \to E * E}, \quad \boldsymbol{E \to (E)}, \quad \boldsymbol{E \to a}, \quad \boldsymbol{E \to b}, \quad \boldsymbol{E \to c}$$

$$\boldsymbol{S = E}$$

$$\boldsymbol{E \Rightarrow E * E \Rightarrow (E) * E \Rightarrow (E + E) * E \Rightarrow (a + E) * E \Rightarrow (a + b) * E \Rightarrow (a + b) * c}$$

$\underline{\boldsymbol{(a + b) * c}}$ **Palabra Generada**

Lenguaje generado por una gramática **G = (V, T, P, S)** al conjunto de cadenas formadas por símbolos terminales y que son derivables a partir del símbolo de partida, es decir:

$$L(G) = \{u \in T^* \mid S \Rightarrow^* u$$

$\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{T}, \boldsymbol{P}, \boldsymbol{S})$, donde $\boldsymbol{V} = \{\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{B}\}$, $\boldsymbol{T} = \{\boldsymbol{a}, \boldsymbol{b}\}$, el símbolo de partida es **S** y las reglas son:

$$\boldsymbol{S \to aB}, \quad \boldsymbol{S \to bA}, \quad \boldsymbol{A \to a}, \quad \boldsymbol{A \to aS}, \quad \boldsymbol{A \to bAA}, \quad \boldsymbol{B \to b}, \quad \boldsymbol{B \to bS}, \quad \boldsymbol{B \to aBB}$$

Ejemplo:

$S \equiv$ cadena con $a's = b's$

$B \equiv$ una b de más

Empieza en: **ab(resto)**

$$\boldsymbol{S \to aB \to ab}$$

Debe ser: **aa(resto)**

$$\boldsymbol{S \to aB \to aaBB}$$

Si queremos que empiece por **b(resto)**

$$\boldsymbol{S \to bA}$$

Si queremos que empiece por **bb(resto)**

$$\boldsymbol{S \to bA \to bbAA}$$

Esta gramática genera el lenguaje: $\boldsymbol{L(G) = \{u \mid u \in \{a, b\}^+ \ \mathbf{y} \ \mathrm{N_a}(u) = N_b(u)\}}$

Donde $\mathrm{N_a}(\boldsymbol{u})$ y $\mathrm{N_b}(\boldsymbol{u})$ son el numero de apariciones de símbolos **a** y **b**, en **u**, respectivamente.

Ejemplo 2:

Sea $\boldsymbol{G} = (\{\boldsymbol{S}, \boldsymbol{X}, \boldsymbol{Y}\}, \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}, \boldsymbol{P}, \boldsymbol{S})$ donde **P** tiene las reglas:

$$\boldsymbol{S \to abc}, \quad \boldsymbol{S \to aXbc}, \quad \boldsymbol{Xb \to bX}, \quad \boldsymbol{Xc \to Ybcc}, \quad \boldsymbol{bY \to Yb}, \quad \boldsymbol{aY \to aaX}, \quad \boldsymbol{aY \to aa}$$

Gramática dependiendo del contexto:

abc pertenece a $\boldsymbol{L(G) = \{a^n b^n c^n\}}$, $\boldsymbol{n \geq 1}$ queremos generar la cadena (**abc**)

$S \to abc$

Si queremos generar la cadena (**aabbcc**)

$S \to aXbc \to abXc \to abYbcc \to aYbbcc \to aabbcc$

Una cadena no valida seria abcabc, porque no cumple con la estructura, que es una serie de **a's** a continuación una de **b's** y otras de **c's**.

De tal forma que esta gramática genera el lenguaje: $\{a^n b^n c^n \mid n = 1, 2, 3, \dots\}$

Tipos de jerarquía de Chomsky:

- **Tipo 0:** Cualquier gramática (**Sin restricciones**), es decir, es hacer básicamente un traductor nuevo. Son lenguajes **recursivamente enumerables**.
- **Tipo 1:** Si todas las producciones tienen la forma $\alpha_1 A \alpha_2 \to \alpha_1 \beta \alpha_2$ donde $\alpha_1, \alpha_2, \beta \in (V \cup T)^*, A \in V$ y $\beta \neq \varepsilon$, excepto posiblemente la regla $S \to \varepsilon$, en cuyo caso **S** no aparece a la derecha de las reglas. **Son lenguajes dependientes del contexto.**
- **Tipo 2:** Si cualquier producción tiene forma $A \to \alpha$ donde $A \in V$, $\alpha \in (V \cup T)^*$**. Son lenguajes independientes del contexto.**
- **Tipo 3:** Si toda regla tiene la forma $A \to uB$ o $A \to u$ donde $u \in T^*$ y $A, B \in V$. **Son conjuntos regulares (Lenguajes Regulares), son Autómatas Finitos Deterministas.**

Conjutos regulares tipo 3: **S->0S, S->B, B->1B, B->ε**

Un lenguaje se dice que es de **tipo i (i=0, 1, 2, 3)** si y solo si es generado por una gramática de **tipo i.**

Demostrar que la gramática $G = (\{S\}, \{a, b\}, \{S \to \varepsilon, S \to aSb\}, S)$ genera el lenguaje

$L = \{a^i b^i \mid i = 0, 1, 2, \dots\}$

Inicialmente tenemos 2 opciones $S \to \varepsilon, S \to aSb$

Con eso generamos la palabra vacía o continuamos generando, de tal forma que:

$S \to aSb \to aaSbb \to aabb$

Si seguimos este procedimiento, nos encontramos que podemos ir generando todas las palabras de la forma $a^i b^i$. De tal forma que son las únicas palabras que se pueden generar.

CAERA EN EL EXAMEN:

$$L = \{uu^{-1} \mid u \in \{a.b\}^*\}$$

De tal forma que este lenguaje genera $u = aab, u^{-1} = baa; uu^{-1} = aabbaa$, esto quiere decir que genera cadenas palíndromas.

Entonces la gramática sería:

$$S \to aSa, \qquad S \to bSb, \qquad S \to \varepsilon$$

$$S \to aSa \to aaSaa \to aabSbaa \to aabbaa$$

Ejercicio:

Determina si la gramática $\boldsymbol{G} = (\{\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{B}\}, \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}\}, \boldsymbol{P}, \boldsymbol{S})$ donde **P** es el conjunto de reglas de producción: $\boldsymbol{S} \to \boldsymbol{AB}, \quad \boldsymbol{A} \to \boldsymbol{Ab}, \quad \boldsymbol{A} \to \boldsymbol{a}, \quad \boldsymbol{B} \to \boldsymbol{cB}, \quad \boldsymbol{B} \to \boldsymbol{d}$

Genera un lenguaje de tipo 3.

Para empezar, $\boldsymbol{S} \to \boldsymbol{AB}, \quad \boldsymbol{A} \to \boldsymbol{Ab}$ no son lenguajes regulares, puesto que solo admite 1 variable y en un orden especifico (la variable siempre a la derecha): ($\boldsymbol{A} \to \boldsymbol{uB}$ o $\boldsymbol{A} \to \boldsymbol{u}$)

Conclusión en este caso estas reglas son **libre del contexto**, de tal forma que este algoritmo **no es eficiente**.

Una forma óptima sería:

$$\boldsymbol{S} \to \boldsymbol{aB}, \quad \boldsymbol{B} \to \boldsymbol{bB}, \quad \boldsymbol{B} \to \boldsymbol{C}, \quad \boldsymbol{C} \to \boldsymbol{cC}, \quad \boldsymbol{C} \to \boldsymbol{d}$$

Esta gramática generaría $\boldsymbol{ab^i c^j d} : \boldsymbol{i}, \boldsymbol{j} \in \mathbb{N}$

Y como esta gramática es de tipo 3, el lenguaje lo es.