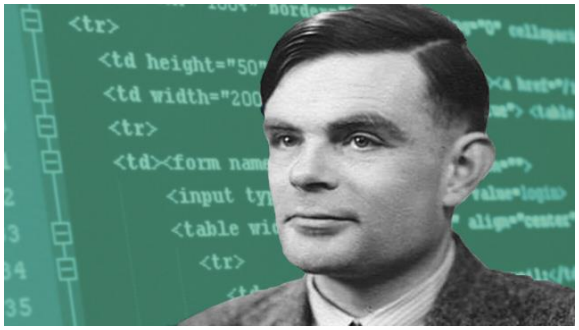
1. Introducción

¿Qué es la teoría de autómatas?

La teoría de autómatas es el estudio de "máquinas abstractas".



Ya en los años 30, antes que existieran las computadoras, Alan Turing estudió las máquinas abstractas, e intentó describir de forma precisa los límites entre lo que una máquina de cálculo podía y no podía hacer.

La investigación de estas máquinas abstractas permitió modelar el funcionamiento de una computadora ideal y también separar aquellos problemas que se podían resolver de forma eficiente mediante una computadora de aquellos que son insolubles (o NP- difíciles)

El estudio de las máquinas abstractas se complementa con el de las gramáticas formales, idea de Noam Chomsky para modelar los lenguajes.

Los autómatas y determinados tipos de gramáticas formales se emplean en el diseño y construcción de software.

Las máquinas de Turing ayudan a comprender lo que podemos esperar de nuestro software.

La teoría de los problemas intratables nos permite deducir si podremos enfrentarnos a un problema y escribir un programa para resolverlo o si, por el contrario, tenemos que hallar una forma diferente de resolverlo, como una aproximación, método heurístico u otra.

Conceptos fundamentales.

Alfabeto

Un *alfabeto* (Σ) es un conjunto finito no vacío de símbolos.

Ejemplos:

1. $\Sigma = \{0,1\}$, el alfabeto *binario*.
2. $\Sigma = \{a, b, c, \ldots z\}$, el alfabeto *de letras minúsculas*.

Cadena

Una *cadena o palabra* es una secuencia finita de símbolos seleccionados de algún alfabeto.

Ejemplo: 00001010 es una palabra dentro del alfabeto *binario*.

Longitud de una cadena

La *longitud* de una cadena es la cantidad de posiciones que ocupan los símbolos que tiene.

Ejemplo: 00001010 es una palabra de longitud 8.

La cadena vacía

La cadena vacía es una cadena de longitud 0.

La denotaremos con:

λ

Aunque en alguna bibliografía también se denota con el símbolo:
ϵ

Potencias de un alfabeto

Si ∑ es un alfabeto, podemos expresar el conjunto de todas las cadenas de una determinada longitud de dicho alfabeto utilizando una notación exponencial.
Es decir:
$\Sigma^k = \{\omega \; con \; símbolos \; en \; \Sigma / \; |\omega| = k\}$

Ejemplo:

Si ∑ = {0,1}

Entonces se tiene:
$\Sigma^0 = \{\lambda\}$
$\Sigma^1 = \{0,1\}$
$\Sigma^2 = \{00,01,10,11\}$
Etc.

Clausuras de un alfabeto

A la unión generalizada de todas las potencias de un alfabeto se la denomina clausura Kleene Σ^*
$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup ... \cup \Sigma^n$

Si se excluye la cadena vacía se denomina clausura positiva Σ^+
$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup ... \cup \Sigma^n$

Concatenación de cadenas

Sean x e y dos cadenas. Entonces, xy denota la concatenación de x e y, es decir, la cadena formada por una copia de x seguida de una copia de y.

Si x es la cadena compuesta por i símbolos
$x = a_1 a_2 ... a_{i-1} a_i$

Y siendo y la cadena compuesta por j símbolos

$y = b_1 b_2 ... b_{j-1} b_j$

Entonces xy es la cadena de i+j símbolos:
$xy = a_1 a_2 ... a_{i-1} a_i b_1 b_2 ... b_{j-1} b_j$

Prefijos y sufijos

Si z está formada por la concatenación de las cadenas x e y, se dice que:

x es ***prefijo*** de z

y es ***sufijo (posfijo)*** de z

Se llama prefijo propio (o posfijo propio) si es distinto de λ

Propiedades de la concatenación:

- **Cerrada:** $\forall x, y \in \Sigma^* : x \cdot y \in \Sigma^*$ (la concatenación de palabras es otra palabra sobre el mismo alfabeto)
- **Asociatividad:** $\forall x, y, z \in \Sigma^* : x \cdot (y \cdot z) = (x \cdot y) \cdot z$
- **Elemento neutro:** $\forall x \in \Sigma^* : \lambda \cdot x = x \cdot \lambda = x$

Reverso de una palabra

Si $w = a_1 a_2 ... a_n$ es una palabra sobre Σ, entonces la palabra **inversa**, o **refleja** de w se define como

$w^R = a_n a_{n-1} ... a_1$

Lenguaje sobre un alfabeto

Un lenguaje sobre Σ es cualquier subconjunto de Σ^*
Es decir:
$L \subseteq \Sigma^*$

Operaciones con lenguajes

Como con cualquier conjunto se pueden hacer las operaciones:

- Unión de lenguajes
- Intersección de lenguajes
- Diferencia de lenguajes

Además, existen las siguientes:

- **Producto de lenguajes o concatenación.**

$$L_1 \cdot L_2 = \{w / w = xy, x \in L_1 \wedge y \in L_2\}$$

- **Potencia de Lenguajes.**

  Consiste en el lenguaje resultante de concatenar el lenguaje consigo i veces.

$$L^i = L \cdot L ... \cdot L (i \text{ veces})$$

- **Clausura de un Lenguaje.**

  La clausura positiva de un lenguaje L se define por: $L^+ = \bigcup_{i=1}^{\infty} L^i$

- **Clausura de Kleene de un lenguaje.**

  La clausura de Kleene se define por: $L^* = \bigcup_{i=0}^{\infty} L^i$

Definiciones recursivas

Las definiciones recursivas tienen un ***caso base***, en el que se definen una o más estructuras elementales, y un ***paso de inducción***, en el que se definen estructuras más complejas en términos de estructuras previamente definidas.

Definición alternativa de cadena

Formalmente las cadenas pueden definirse **inductivamente** a partir de una constante – la cadena vacía – y de un constructor - **·** - que permite agregar *un símbolo* a la **cabeza** de una cadena.
Es decir:
Definición de cadena:

- **Base**: λ es una cadena.
- **Paso inductivo**: si $t \in \Sigma$ y $\omega \in \Sigma^*$ ya es una cadena, entonces $t\omega$ es una cadena.

Definición alternativa de potencia de un alfabeto

Se puede definir Σ^i inductivamente como:

- **BASE:** Si i = 0, entonces $\Sigma^i = \{\lambda\}$
- **PASO INDUCTIVO:** Si i > 0, entonces $\Sigma^i = \{a \cdot \alpha / a \in \Sigma \wedge \alpha \in \Sigma^{i-1}\}$

Definición alternativa de Reverso de una palabra

- **Base**: el reverso de λ es λ.
- **Paso inductivo**: si $t \in \Sigma$ y $\omega \in \Sigma^*$, entonces $(t\omega)^r = (\omega)^r t$.

Inducción Estructural

Cuando una estructura fue definida recursivamente, se pueden probar teoremas acerca de ella utilizando la siguiente forma de demostración, que recibe el nombre de ***inducción estructural***.

Sea S(X) una proposición acerca de estructuras X definidas mediante alguna definición recursiva determinada.

1. como **base**, se prueba $S(X)$ para la(s) estructura(s) base X
2. para el **paso de inducción**, se toma una estructura X , que, según la definición recursiva, está formada a partir de $Y_1, Y_2, ..., Y_k$, se dan por ciertas las proposiciones $S(Y_1), S(Y_2), ...S(Y_k)$ y se utilizan para probar $S(X)$

Ejemplo: *"El número de nodos de un árbol es superior al de arcos en una unidad."*
Definición recursiva de un **ÁRBOL:**

- **Base**: un único nodo es un árbol.
- **Paso inductivo**: si $T_1, T_2, ..., T_k$ son árboles, se puede construir un nuevo árbol de la siguiente manera:
  1. se comienza con un nuevo nodo (N), que es la raíz del árbol.
  2. se añaden copias de todos los árboles $T_1, T_2, ..., T_k$
  3. se añaden arcos desde el nodo N hasta las raíces de cada uno de los árboles $T_1, T_2, ..., T_k$

Prueba: "Si T es un árbol y T tiene n nodos, y e arcos, entonces $n = e + 1$"

Base: El caso base ocurre cuando T es un único nodo. Entonces hay 0 arcos y se cumple 1 = 0 + 1, así que $n = e + 1$ es verdadera

Paso inductivo: Sea T un árbol construido de la forma indicada previamente. Se da por cierto que la propiedad se cumple para $T_1, T_2, \ldots, T_k$. Entonces, para cada uno, $n_i = e_i + 1$.
El total de nodos del nuevo árbol es igual a $n = n_1 + n_2 + \ldots + n_k + 1$ (todos los nodos de cada árbol y el nodo raíz)
El total de arcos del nuevo árbol es igual a $e = e_1 + e_2 + \ldots + e_k + k$ (todos los arcos de cada árbol y los que unen el nodo raíz con cada árbol)
La propiedad se cumple para $T_1, T_2, \ldots, T_k$ así que el total de nodos del nuevo árbol es
$(e_1 + 1) + (e_2 + 1) + \ldots + (e_k + 1) + 1 = e_1 + e_2 + \ldots + e_k + k + 1$, o sea, $e + 1$

Gramáticas.

Una **gramática** es un sistema matemático para definir un lenguaje.
Define la estructura de las palabras de un lenguaje.

Definición de una gramática

Es una 4-tupla $G = (N, \Sigma, P, S)$, donde:

- N es un conjunto finito de símbolos ***no terminales*** (variables o categorías sintácticas)
- Σ es un conjunto finito de símbolos ***terminales***, $N \cap \Sigma = \varnothing$
- Un elemento (α, β) en P se escribe $\alpha \to \beta$ y se llama ***producción***.
- S es un símbolo distinguido en N , llamado sentencia o ***símbolo inicial***.

Forma sentencial

La ***forma sentencial*** de una gramática es una cadena especial que se define recursivamente de la siguiente manera:

Base: S es una *forma sentencial*.

Paso inductivo:

Si $\alpha\beta\gamma$ es una forma sentencial y $\beta \to \delta \in P$, entonces $\alpha\delta\gamma$ también es una *forma sentencial*.

Una forma sentencial de G que sólo contiene símbolos terminales o es la palabra vacía se llama ***sentencia*** generada por G.

Derivación

- ***Derivación(*** $\underset{G}{\Rightarrow}$ ***)*** Si $\alpha\beta\gamma$ es una cadena en $(N \cup \Sigma)^*$ y $\beta \to \delta$ es una producción, entonces $\alpha\beta\gamma \underset{G}{\Rightarrow} \alpha\delta\gamma$
- ***Derivación **** (0 o más pasos)

Lenguaje generado por una gramática

Un ***lenguaje generado por una gramática*** G **, *es decir*** , $L(G)$ es el conjunto de ***sentencias*** generadas por G

Usando el concepto de derivación:

$$L(G) = \{\omega \in \Sigma^* / S \underset{*}{\Rightarrow} \omega\}$$

Clasificación de gramáticas (Jerarquía de Chomsky)

Las gramáticas pueden clasificarse según el formato de sus producciones.

Si un lenguaje es generado por una gramática de tipo X, ,entonces el lenguaje es de tipo X.

- **Gramáticas tipo 0**

  Son las más generales. (**Gramáticas sin restricciones**).Las producciones pueden ser de cualquier tipo permitido, es decir de la forma $\alpha \to \beta$ con $\alpha \in (N \cup \Sigma)^+$ y $\beta \in (N \cup \Sigma)^*$

  Los lenguajes generados por estas gramáticas son los ***Lenguajes con Estructura de Fras***e (se agrupan en la clase L_0) Estos lenguajes también se conocen como ***lenguajes recursivamente enumerables***.

- **Gramáticas tipo 1**

  Gramáticas sensibles al contexto.

  Las producciones son de la forma $\alpha \to \beta$ con $|\alpha| \leq |\beta|$, excepto para $S \to \lambda$.

  Los lenguajes generados por estas gramáticas son los l**enguajes sensibles al contexto**. (L_1)

- **Gramáticas tipo 2**

  Gramáticas libres del contexto.

  Las producciones son de la forma $A \to \alpha$.

  Los lenguajes generados por estas gramáticas son los **lenguajes libres de contexto** (L_2).

- **Gramáticas tipo 3**

  Gramáticas regulares.

  Lineales por derecha:

  $$A \to bC$$
  $$A \to b$$
  $$A \to \lambda$$

  Lineales por izquierda:

  $$A \to Cb$$
  $$A \to b$$
  $$A \to \lambda$$

  Los lenguajes generados por estas gramáticas son los **lenguajes regulares** (L_3)

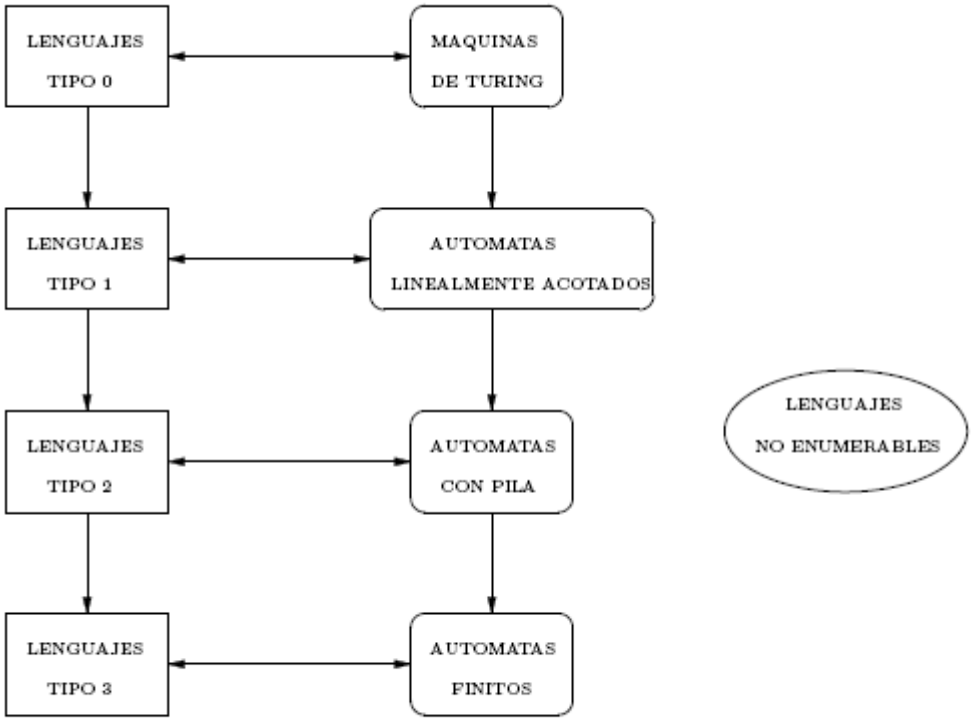


Figura 1: Relación Lenguajes-Máquinas Abstractas