Lenguajes Formales, Autómatas y Computabilidad

Clase Teórica

Gramáticas

Segundo Cuatrimestre 2024

Bibliografía

Capítulo 1, *Formal Languages and theor relation to automata* J. Hopcroft, J. Ullman, Addison Wesley, 1969

Introduction to Automata Theory, Languages and Computation, J. Hopcroft, R. Motwani, J. Ullman, Second Edition, Addison Wesley, 2001.

En esta clase

- Gramáticas
- Lenguaje generado por gramáticas
- jerarquía de Chomsky
- Cotas en el árbol de derivación

Alfabetos y Lenguaje

Un alfabeto es un conjunto finito, no vacío, de símbolos.

Consideramos la concatenación de símbolos.

Una palabra sobre el alfabeto Σ es una secuencia finita de elementos (cero o más) de un alfabeto. Tamién las llamamos palabras o strings.

palabra nula λ. No tiene símbolos.

Ejemplos: Estas son algunas palabras sobre $\Sigma = \{a, j, r\}$,
a, j, aa, aj, ar, ja, jj, aaj, rja, $raja$, $jarra$, etc.

El conjunto de todas las plabras sobre un alfabeto

Dado alfabeto Σ, la clausura de Kleene del alfabeto Σ, que denotamos Σ^* se define así:

$\Sigma^0 = \{\lambda\}$

$\Sigma^1 = \Sigma = \{a : a \in \Sigma\}$

$\Sigma^2 = \Sigma\Sigma = \{ab : a \in \Sigma, b \in \Sigma\}$

...

$\Sigma^* = \bigcup_{i \geq 0} \Sigma^i$

El conjunto de todas las plabras sobre un alfabeto

¿Cuál es la cardinalidad del conjunto Σ^i?

El conjunto Σ tiene $|\Sigma|$ elementos.
Para cada $i \geq 0$, Σ^i tiene $|\Sigma|^i$ palabras.
El conjunto Σ^* tiene una cantidad infinita numerable de palabras.

Clausura positiva

$$\Sigma^+ = \bigcup_{i \geq 1} \Sigma^i$$

Hay tantas palabras como números naturales

Teorema

$|\Sigma^*|$ es igual al la cardinalidad de $\mathbb{N}$.

Un orden entre pares de elementos es una relación antisimétrica y transitiva.

Definición (Orden longitud-lexicográfico en Σ^*)

Definimos el orden $\prec \subset \Sigma^* \times \Sigma^*$:
Asumimos un orden lexicográfico entre los elementos del alfabeto.
Lo extendemos a un lexicografico entre todas las palabras de la misma longitud.
Las palabras de menor longitud son menores que las de mayor longitud.

Por ejemplo para $\Sigma = \{a, b, c\}$,
$\lambda \prec a \prec b \prec c \prec aa \prec ab \prec ac \prec ba \prec bb \prec bc\ ldots$.

Demostración del Teorema.

Definimos una biyección $f : \mathbb{N} \to \Sigma^*$, $f(i) =$ la i-ésima palabra en el orden $\prec$ longitud lexicografico sobre Σ^*. $\square$

Lenguaje sobre un alfabeto

Un lenguaje L sobre un alfabeto Σ es un conjunto de palabras sobre Σ.
Es decir, $L \subseteq \Sigma^*$.

Ejemplos:
$\emptyset$
$\{\lambda\}$ (notar que es distinto de $\emptyset$)
$\{0, 01, 011, 0111, 01111, \ldots\ldots\}$, es un lenguaje sobre $\Sigma = \{0, 1\}$.

¿Cuántos lenguajes hay?

Definición

Si A *es un conjunto,* $\mathcal{P}(A)$ *es el conjunto de todos los subconjuntos de* A*,* $\mathcal{P}(A) = \{B \subseteq A\}$*.*

Si A es un conjunto finito $\mathcal{P}(A) = 2^{|A|}$.

Ejemplo: $A = \{a, b, c\}$ entonces $|\mathcal{P}(A)| = 2^3$,
$\mathcal{P}(A) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$.

Ejemplo: Σ^* es el conjunto de todas las palabras sobre Σ, entonces $|\mathcal{P}(\Sigma^*)| = 2^{\mathbb{N}}$.

La cantidad de lenguajes es no numerable

El conjunto de todos los lenguajes sobre alfabeto Σ es .

$$\mathcal{P}(\Sigma^*) = \{L \subseteq \Sigma^*\}$$

Teorema

$|\Sigma^*| < |\mathcal{P}(\Sigma^*)|$.

Demostración.

Supongamos que $\mathcal{P}(\Sigma^*)$ es numerable y supongamos el orden longitud y lexicográfico en cada L_i,

$L_1 : w_{1,1}, w_{1,2}, w_{1,3}, w_{1,4}, w_{1,5} \ldots$
$L_2 : w_{2,1}, w_{2,2}, w_{2,3}, \ldots$
$L_3 : w_{3,1}, w_{3,2}, w_{3,3}, \ldots$
$\vdots$

Definamos $L = \{u_1, u_2, \ldots\}$ donde $u_1 < u_2 < \ldots$ y para todo i, $w_{i,i} < u_i$.
Entonces L no es ninguno de los L_i, para todo i, porque la i-ésima palabra en L en el orden $<$ es u_i, que es distinta a $w_{i,i}$.
Entonces $|\Sigma^*| < |\mathcal{P}(\Sigma^*)|$.

$\square$

Lenguaje formales y su tratamiento computacional

¿Todos los lenguajes se pueden reconocer/generar computacionalmente?

¿Cuál es la dificultad computacional ?

¿Qué es una computadora?

Un autómata es un dispositivo que realiza pasos mecánicamente o electrónicamente, de manera autómatica.
Vistos como modelos de cómputo, hay autómatas con distintas capacidades.
autómatas finitos
autómatas de pila
máquinas de Turing
maquinas probabilsíticas
máquinas cuánticas

y hay más

A lo largo del curso definiremos los 3 primeros de la lista y mostraremos su capacidades.
¿Cuántos autómatas hay?
Tan solo una cantidad numerable, ya que son definidos finitariamente.
Se los puede ver a todos como programas

¿Todos los lenguajes se tratan computacionalmente?

No todos. Solamente una cantidad numerable.

Hay una cantidad no numerable de lenguajes ($|\mathcal{P}(\Sigma^*|)$ y hay tan solo una cantidad numerable de procedimientos computacionales.

Los estudaremos según la jerarquía de Chomsky, que se corresponde con una jerarquía de los autómatas capaces de reconocerlos.

Relaciones

Dados los conjuntos A y B, se llama relación de A en B a cualquier subconjunto de $A \times B$.

Una relación $R \subseteq A \times A$ es reflexiva cuando

$$\forall a \in A, (a, a) \in R.$$

Ejemplo: "$\leq$"sobre $\mathbb{N}$.

Una relación $R \subseteq A \times A$ es simétrica cuando

$$\forall a, b \in A, \Big(\text{Si } (a, b) \in R \text{ entonces } (b, a) \in R\Big).$$

Ejemplo: "$\neq$"sobre $\mathbb{N}$.

Una relación $R \subseteq A \times A$ es transitiva cuando

$$\forall a, b, c \in A, \Big(\text{Si } (a, b) \in R \wedge (b, c) \in R \text{ entonces } (a, c) \in R\Big).$$

Ejemplo: "a paralela a b", en el conjunto de rectas del plano.

Una relación es de equivalencia, cuando es reflexiva, simétrica y transitiva.

Composición de relaciones

Sean A, B y C tres conjuntos, y sean R y G dos relaciones tales que $R \subseteq A \times B$ y $G \subseteq B \times C$.

La relación de composición: $G \circ R \subseteq A \times C$ se define como

$$G \circ R = \left\{ (a, c) \, , a \in A, c \in C : \exists b \in B \text{ tal que } aRb \wedge bGc \right\}.$$

Una relación R definida sobre A es de identidad (id_A) si se cumple que

$$\forall a, b \in A, \; a \, id_A \, b \text{ si y solo si } a = b.$$

La relación de identidad es el elemento neutro de la composición. Dada una relación $R \subseteq A \times B$ es cierto que

$$id_B \circ R = id_A \circ R$$

Relación potencia

Dada una relación $R \subseteq A \times A$, y dado n se define la potencia $R^n \subseteq A \times A$ como

$$R^n = \begin{cases} id_A & \text{si } n = 0 \\ R \circ R^{n-1} & \text{si no} \end{cases}$$

con $R = R^1$.

Notar que R^n es un conjunto de pares, cualquiera sea el valor de n.

Clausura transitiva

Dada una relación $R \subseteq A \times A$ se define clausura transitiva R^+,

$$R^+ = \bigcup_{k=1}^{\infty} R^k.$$

Proposición

1. $R \subseteq R^+$
2. R^+ *es transitiva*
3. *Si* $S \subseteq A \times A$, $R \subseteq S$ *y* S *es transitiva entonces* $R^+ \subseteq S$.

Entonces, R^+ es la relación transitiva más pequeña que contiene a R.

Demostración de la proposición

Queremos ver que si aR^+b y bR^+c enonces aR^+c.

Si aR^+b, entonces existe una secuencia de elementos $d_1, \ldots, d_n$ tal que $d_1Rd_2, \ldots, d_{n-1}Rd_n$, donde $d_1 = a$ y $d_n = b$. Por lo tanto, aR^nb. Análogamente, como bR^+c entonces existe una secuencia de elementos $e_1, \ldots, e_m$ tal que $e_1Re_2, \ldots, e_{m-1}Re_m$, donde $e_1 = b$ y $e_m = c$. Por lo tanto bR^mc. Concluimos que $aR^{n+m}c$ y esto implica aR^+c.

Ahora demostremos que si $R \subseteq S$ y S es transitiva entonces $R^+ \subseteq S$. Supongamos aR^+b. Entonces, existe una secuencia de elementos $c_1, \ldots, c_n$ tal que $c_1Rc_2, \ldots, c_{n-1}Rc_n$, donde $c_1 = a$ y $c_n = b$. Como $R \subseteq S$ tenemos que $c_1Sc_2, \ldots, c_{n-1}Sc_n$, y como S es transitiva entonces, la aplicación repetida de la transitividad nos lleva a que c_1Sc_n, o sea aSb.
$\square$

Clausura transitiva reflexiva: R^*

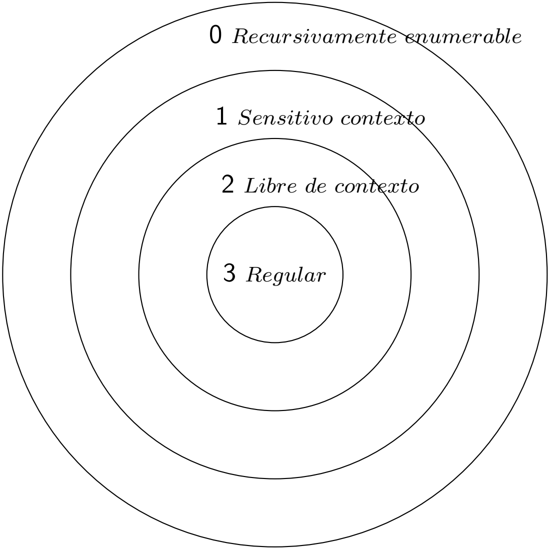
$$R^* = R^+ \cup id = \bigcup_{i=0}^{\infty} R^i.$$

La jerarquía de Chomsky (Noam Chomsky en 1956).

Es una clasificación jerárquica de tipos de gramáticas formales que generan lenguajes formales.

Noam Chomsky

La jerarquía de Chomsky



Gramáticas

Definición

Una gramática es una 4-upla $G = \langle V_N, V_T, P, S \rangle$ donde

- V_N *es un conjunto de símbolos llamados no-terminales*
- V_T *es un conjunto de símbolos terminales (tal como lo era Σ en los ejemplos anteriores)*
- P *es el conjunto de "producciones", que es un conjunto finito de*

  $$(V_N \cup V_T)^* \, V_N \, (V_N \cup V_T)^* \times (V_N \cup V_T)^* \, ,$$

  estas producciones son entonces pares ordenados (α, β), que usualmente son notados como $\alpha \to \beta$.
- $S \in V_N$ *es el símbolo distinguido de V_N.*

Ejemplo

Sea $G = \langle V_N, V_T, P, S\rangle$ la gramática libre de contexto tal que $V_N = \{S\}$, $V_T = \{+, *, \mathbf{a}, (,)\}$, y P tiene las producciones

$$\begin{aligned} &S \to S + S, \\ &S \to S * S, \\ &S \to (S)\,, \\ &S \to \mathbf{a} \end{aligned}$$

Lenguaje generado por una gramática

Definición (Lenguaje generado de una gramática G)

Dada una gramática $G = (V_N, V_T, P, S)$*,*

$$\mathcal{L}(G) = \{w \in V_T^* : S \underset{G}{\overset{+}{\Rightarrow}} w\}$$

donde $\underset{G}{\overset{+}{\Rightarrow}}$ *es derivación en uno o más pasos, que se obtiene de la clausura transitiva de la derivación directa* $\underset{G}{\Rightarrow}$*. Damos a continuación la definición.*

Ejemplo

Sea $G = \langle V_N, V_T, P, S \rangle$ la gramática libre de contexto tal que
$V_N = \{S\}$,
$V_T = \{+, *, \mathbf{a}, (,)\}$, y
P tiene las producciones

$$
\begin{aligned}
S &\rightarrow S + S, \\
S &\rightarrow S * S, \\
S &\rightarrow (S)\,, \\
S &\rightarrow \mathbf{a}
\end{aligned}
$$

$S \overset{+}{\Rightarrow} (\mathbf{a})$, porque $S \Rightarrow (S) \Rightarrow (\mathbf{a})$

Si elegimos el no-terminal más a la izquierda,
$S \overset{+}{\Rightarrow} (\mathbf{a} + \mathbf{a})$ porque
$S \Rightarrow (S) \Rightarrow (S + S) \Rightarrow (\mathbf{a} + S) \Rightarrow (\mathbf{a} + \mathbf{a})$.

$S \overset{+}{\Rightarrow} (\mathbf{a} + \mathbf{a})$ también si elegimos el no terminal más a la derecha,
$S \Rightarrow (S) \Rightarrow (S + S) \Rightarrow (S + \mathbf{a}) \Rightarrow (\mathbf{a} + \mathbf{a})$.

Definición (Forma sentencial de una gramática)

Sea $G = \langle V_N, V_T, P, S \rangle$.

- S *es una forma sentencial de* G.
- *Si* $\alpha\beta\gamma$ *es una forma sentencial de* G, *y* $(\beta \to \delta) \in P$, *entonces* $\alpha\delta\gamma$ *es también una forma sentencial de* G.

Las formas sentenciales están pertenecen a $(V_N \cup V_T)^*$.

Definición (Derivación directa $\underset{G}{\Rightarrow}$)

Si $\alpha\beta\gamma \in (V_N \cup V_T)^*$ *y* $(\beta \to \delta) \in P$ *entonces* $\alpha\beta\gamma$ *deriva directamente en* G $\alpha\delta\gamma$. *Lo escribimos así:*

$$\alpha\beta\gamma \underset{G}{\Rightarrow} \alpha\delta\gamma.$$

Entonces, $\underset{G}{\Rightarrow}$ *es una relación sobre* $(V_N \cup V_T)^*$, *es decir,*

$$\underset{G}{\Rightarrow} \subseteq (V_N \cup V_T)^* \times (V_N \cup V_T)^* .$$

Podemos componer la relación $\underset{G}{\Rightarrow}$ consigo misma, 0 o más veces...

Clausura de Kleene de la relación de derivación $\underset{G}{\Rightarrow}$

$$\left(\underset{G}{\Rightarrow}\right)^0 = id_{(V_N \cup V_T)^*}$$

$$\text{Si } k > 0, \left(\underset{G}{\Rightarrow}\right)^k = \left(\underset{G}{\Rightarrow}\right)^{k-1} \circ \underset{G}{\Rightarrow}$$

$$\left(\underset{G}{\Rightarrow}\right)^+ = \bigcup_{k=1}^{\infty} \left(\underset{G}{\Rightarrow}\right)^k$$

$$\left(\underset{G}{\Rightarrow}\right)^* = \left(\underset{G}{\Rightarrow}\right)^+ \cup id_{(V_N \cup V_T)^*}$$

Según como se elige el símbolo no terminal a tratar,

Derivación más a la izquierda $\underset{L}{\Rightarrow}$
Derivación más a la derecha $\underset{R}{\Rightarrow}$

Paar hacer i pasos de una derivación escribimos, respectivamente, $\overset{i}{\underset{L}{\Rightarrow}}, \overset{i}{\underset{R}{\Rightarrow}}$.
Y por último tenemos las respectivas clausuras transitivas $\overset{+}{\underset{L}{\Rightarrow}}, \overset{+}{\underset{R}{\Rightarrow}}$ y las respectivas clausutras de Kleene $\overset{*}{\underset{L}{\Rightarrow}}, \overset{*}{\underset{R}{\Rightarrow}}$

La jerarquía de Chomsky

Gramáticas de tipo 0 (gramáticas sin restricciones)
$\alpha \to \beta$,

Gramáticas de tipo 1 (gramáticas sensibles al contexto)
$\alpha \to \beta$, con $|\alpha| \leq |\beta|$

Gramáticas de tipo 2 (gramáticas libres de contexto)
$A \to \gamma$ con $A \in V_N$.

Gramáticas de tipo 3 (gramáticas regulares).
$A \to a$, $A \to aB$, $A \to \lambda$ con $A, B \in V_N, a \in V_T$.

La jerarquía de gramáticas da origen a la jerarquía de los lenguajes:

recursivamente enumerables,
sensitivos al contexto,
libres de contexto y
regulares.

Árbol de derivación gramáticas libres de contexto (y regulares)

Asumamos una gramática $G = (V_N, V_T, P, S)$.

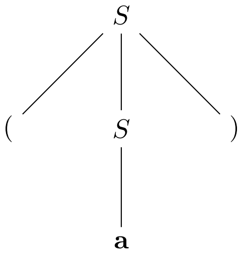
Definición (Árbol de derivación)

Un árbol de derivación es una representación gráfica de una derivación.

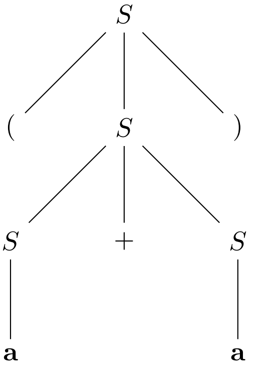
Las etiquetas de las hojas están en $V_T \cup \{\lambda\}$.
Las etiquetas de los nodos internos están en V_N. Las etiquetas de sus hijos son los símbolos del cuerpo de una producción (lado derecho).

Un nodo tiene etiqueta A y tiene n descendientes etiquetados X_1, X_2, $\ldots X_n$, exactamente cuando hay una derivación que usa una producción $A \rightarrow X_1 X_2 \ldots X_n$.

Ejemplo árbol de derivación y derivación más a la izquierda



$$S \underset{L}{\Rightarrow} (S) \underset{L}{\Rightarrow} (a)$$



$$S \underset{L}{\Rightarrow} (S) \underset{L}{\Rightarrow} (S + S) \underset{L}{\Rightarrow} (\mathbf{a} + S) \underset{L}{\Rightarrow} (\mathbf{a} + \mathbf{a})$$

$$C \to CP|P$$
$$P \to (C)|()$$

$$S \to aSb|\lambda$$

$$S \to A$$
$$A \to aAa$$
$$A \to bAb$$
$$A \to c$$

$$S \to aS|bS|\lambda$$

$$S \to \lambda|aA$$
$$A \to a|aB$$
$$B \to aA$$

$$S \to \lambda|Ca$$
$$C \to a|Da$$
$$D \to Ca$$

Lema crucial

Considlares las gramáticas regulares en su forma lineal a derecha, $A \to aB|a|\lambda$.

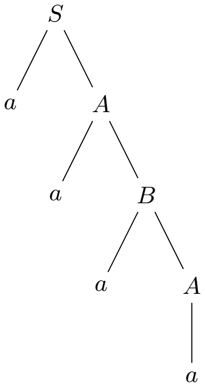
Lema (Lema 4.1 Aho-Ullman vol. 1)

Sea $G = (N, T, P, S)$ *regular, no recursiva a izquierda. Si* $A \overset{i}{\underset{L}{\Rightarrow}} wB$ *entonces* $i = |w|$.

Demostración del Lema

Consideremos el arbol de derivación de $w = a_1 \ldots a_n$
Si cortamos el arbol en altura i, para $i \geq 1$ obtenemos un subarbol con i hojas, $a_1 \ldots a_i$, y el único nodo que no es una hoja tiene como etiqueta un símbolo de V_N. Por la forma de las producciones, cada derivación pone un símbolo terminal. En n derivaciones se producen los n símbolos termninales y el no terminal a la derecha. $\square$

$S \to \lambda | aA \qquad A \to a | aB, \qquad B \to aA$



Lema crucial

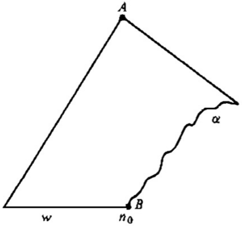
Una gramática libre de contexto no es recursiva a izquierda si no tiene derivaciones $A \underset{L}{\overset{+}{\Rightarrow}} A\alpha$, para $\alpha \in (V_N \cup V_T)^*$.

Lema (Lema 4.1 Aho-Ullman vol. 1)

Sea $G = (N, T, P, S)$ libre de contexto, no recursiva a izquierda. Existe una constante c tal que si $A \underset{L}{\overset{i}{\Rightarrow}} wB\alpha$ entonces $i \leq c^{|w|+2}$.

Demostración del Lema

Llamemos $k = |V_N|$. Consideremos el ábol de la derivación más a la izquierda para $A \overset{i}{\underset{L}{\Rightarrow}} wB\alpha$.



Sea n_0 el nodo con etiqueta B en la derivación $A \overset{i}{\underset{L}{\Rightarrow}} wB\alpha$. Por ser la derivación más a la izquierda, todos los caminos a la derecha del camino desde la raíz a n_0 son más cortos, o del mismo largo. Sea $A = X_0 \frown X_1 \frown X_{(n+2)k} = B$. el camino desde la raíz a n_0, Separemoslos en $(n+2)$ segmentos de $k+1$ nodos,

$$A = X_0 \frown \ldots \frown X_k, \quad X_k \frown \ldots \frown X_{2k}, \ldots, \quad X_{(n+1)k} \frown \ldots \frown X_{(n+2)k} = B.$$

notar que se repite el final de uno con el principio del siguiente.

Cosideremos los $(n+2)$ subarboles de derivación (con β apropiados)

$$A = X_0 \underset{L}{\overset{k}{\Rightarrow}} \beta_1 X_k, \quad X_k \underset{L}{\overset{k}{\Rightarrow}} \beta_2 X_{2k}, \quad \ldots, X_{(n+1)k} \underset{L}{\overset{k}{\Rightarrow}} \beta_{(n+2)k} X_{(n+2)k}.$$

Es imposible que cada uno produzca uno o más símbolos de wB, porque $|wB| = n+1$. Al menos uno no produce ningun símbolo de wB,

$$X_{jk} \underset{L}{\Rightarrow} \beta_{jk+1} X_{jk+1} \underset{L}{\Rightarrow} \ldots \underset{L}{\Rightarrow} \beta_{(j+1)k} X_{(j+1)k}$$

y deriva solamente λ. Cada uno de $X_{jk}, \beta_{jk+1} X_{jk+1}, .., \beta_{(j+1)k} X_{(j+1)k}$ empiezan con un símbolo de V_N, son en total $k+1$, Necesariamente hay dos que empiezan con el mismo símbolo. Pero esto contradice que la gramática no es recursiva a izquierda. Entonces nuestra suposición de que el camino de la raíz a n_0 tiene $(n+2)$ segmentos de $(k+1)$ nodos es imposible. Concluimos que su longitud es menor que $(n+2)k$.
Sea ℓ el máximo número de símbolos en la parte derecha de una producción de la gramática. La cantidad de nodos del árbol de derivación es a lo sumo $\ell^{k(n+2)}$. Por lo tanto, $A \underset{L}{\overset{i}{\Rightarrow}} wB\alpha$ con $i \leq \ell^{k(n+2)}$.

Para finalizar la demostración basta tomar $c = \ell^k$. □

Lenguajes formales y complejidad computacional

Hay lenguajes que se aceptan/generan en tiempo lineal en el tamaño de la entrada (esta es la mínima complejidad posible cuando hay que leer la entrada). Estos son los lenguajes regulares y algunos lenguajes libres de contexto.

El la aceptación/generación de los demás lenguajes libres de contexto usa tiempo polinomial en el tamaño de la entrada.

Y para lenguajes arbitrarios no hay cota en la complejidad computacional de analizarlos.

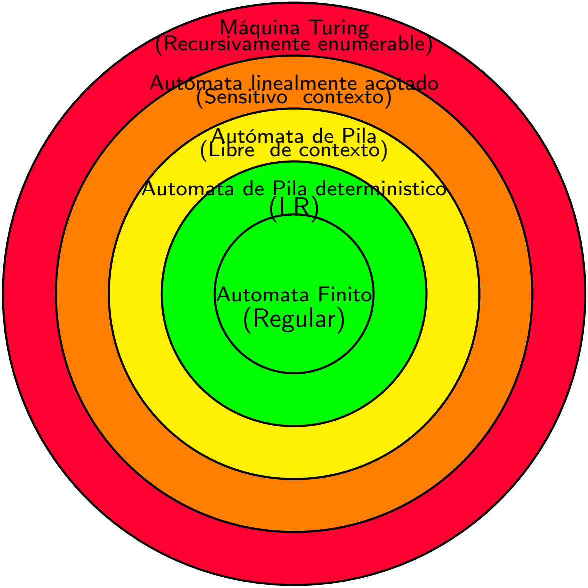
Recordemos las inclusiones conocidas de teoría de complejidad.

$\mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PSPACE} = \mathsf{NPSPACE} \subseteq \mathsf{EXPTIME} \subseteq \mathsf{EXPSPACE}$

L:	espacio logaritimico en máquina Turing determinística
NL:	espacio logaritmico en máquina Turing no determinística
P:	tiempo polinomial en máquina Turing determinística
NP:	tiempo polinomial en una máquina Turing no determinística
PSPACE:	espacio polinomial en máquina Turing determinística

Teorema de Savitch's muestra que PSPACE = NPSPACE

Jeraquía de Lenguajes Formales y su complejidad de aceptación/generación



Complejdad
Verde: lineal
Amarillo: cúbica
Naranja: PSPACE
Rojo: PSPACE