# Introducció

L'autòmat finit és un model formal d'un sistema que treballa amb entrades i eixides discretes. El sistema pot estar en qualsevol de les configuracions d'un conjunt finit de configuracions internes o *estats*. L'estat del sistema resumeix la informació relativa a les anteriors entrades necessària per a determinar el comportament del sistema en les subsegüents entrades.

A banda de l'interés teòric de l'estudi dels autómats finits, ja que modelitzen una de les famílies de llenguatges de la jerarquia de Chomsky, la família dels *Llenguajes Regulars*, també hi ha importants raons pràctiques per al seu estudi, degut a la seua aplicabilitat al disseny de certs tipus d'algorismes molt utilitzats en informàtica. Per exemple, la fase d'*anàlisi lèxica* d'un compilador està sovint basada en la simulació d'un autòmat finit. L'analitzador lèxic analitza els símbols d'un programa font per a localitzar les cadenes de caràcters que corresponen a identificadors, constants numèriques, paraules reservades, etc. En aquest procés l'analitzador lèxic només necessita recordar una quantitat finita d'informació.

La teoria d'autòmats finits és també molt útil en el disseny d'eficients processadors de cadenes. Un exemple senzill és l'ús d'un autòmat finit per a resoldre el problema de detectar en quines posicions apareix una o un conjunt de cadenes distintes (usualment denominades patrons) en una cadena més llarga $x$ (text). Aquest problema es coneix com *String Matching* o *Pattern Matching*.

En aquesta pràctica anem a estudiar la representació dels tres tipus d'autòmat finit, i la implementació de l'anàlisi de cadenes sobre els tres tipus d'autòmats.

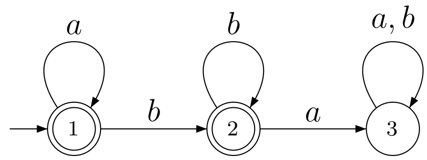
# La representació d'Autòmats Finits

Representarem un Autòmat Finit $A = (Q, \Sigma, \delta, q_0, F)$ com una llista $aut = \{est, alf, trans, ini, fin\}$ amb cinc components que són:

- $est$ una llista amb els estats del *conjunt d'estats* $Q$ (enters, símbols, llistes, etc.).
- $alf$ una llista amb els símbols de l'*alfabet* $\Sigma$.
- $trans$ una llista de transicions que representarà la *funció de transició* $\delta$.

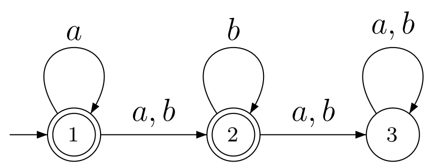
  Cada transició és una llista de tres components $\{EstOrigen, Simbol\ o\ \{\}, EstDestinacio\}$.
- $ini$ es el nom de l'*estad inicial* $q_0$.
- $fin$ una llista amb el *conjunt d'estats finals* $F$.

# Exemples



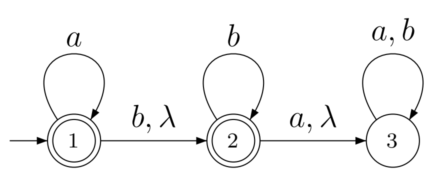

En *Mathematica*:

$A = \{\{1, 2, 3\}, \{a, b\}, \{\{1, a, 1\}, \{1, b, 2\}, \{2, a, 3\}, \{2, b, 2\}, \{3, a, 3\}, \{3, b, 3\}\}, 1, \{1, 2\}\}$




En *Mathematica*

$$A = \{\{1, 2, 3\}, \{a, b\},$$
$$\{\{1, a, 1\}, \{1, a, 2\}, \{1, b, 2\}, \{2, a, 3\}, \{2, b, 3\}, \{2, b, 2\}, \{3, a, 3\}, \{3, b, 3\}\},$$
$$1, \{1, 2\}\}$$



En *Mathematica*

$$A = \{\{1,2,3\},\{a,b\},$$
$$\{\{1,a,1\},\{1,\{\},2\},\{1,b,2\},\{2,a,3\},\{2,\{\},3\},\{2,b,2\},\{3,a,3\},\{3,b,3\}\},$$
$$1,\{1,2\}\}$$

# Exercicis

**Exercici 1**

Es demana implementar un mòdul *Mathematica* que, prenent un AF $A$ com entrada, torne *True* si $A$ és determinista i *False* en cas contrari.

**Exercici 2**

Donat un $AFD$ direm que és bideterminista si té un únic estat final i no conté dues transicions que amb el mateix símbol arriben al mateix estat. Es demana implementar un mòdul *Mathematica* que, prenent com entrada un $AFD$ $A$, torne *True* si $A$ és bideterminista i *False* en cas contrari.

**Exercici 3**

Es demana implementar un mòdul *Mathematica* que, prenent un $AFD$ $A$ com entrada, torne *True* si $A$ està completament especificat i *False* en cas contrari.

**Exercici 4**

Es demana implementar un mòdul *Mathematica* que, prenent un $AFD$ $A$ i una cadena $x$ com entrada, torne *True* si la cadena és acceptada per l'autòmat i *False* en cas contrari.

**Exercici 5**

Es demana implementar un mòdul *Mathematica* que, prenent dos $AFD$s $A_1$ i $A_2$, i una cadena $x$ com entrada, torne *True* si la cadena pertany a $L(A_1) \cup L(A_2)$ i *False* en cas contrari.

**Exercici 6**

Es demana implementar un mòdul *Mathematica* que, prenent dos $AFD$s $A_1$ i $A_2$, i una cadena $x$ com entrada, torne *True* si la cadena pertany a $L(A_1) \cap L(A_2)$ i *False* en cas contrari.

**Exercici 7**

Es demana implementar un mòdul *Mathematica* que, prenent un $AFN$ $A = (Q, \Sigma, \delta, q_0, F)$, un conjunt $C \subseteq Q$ i un símbol $a \in \Sigma$ com entrada, torne $\delta(C, a)$ (el conjunt d'estats resultat d'analitzar en l'autòmat $A$ el símbol $a$ a partir dels estats en $C$).

**Exemple:** Donats el AFN:

$$\begin{aligned} A = \{&\{1,2,3\},\{a,b\},\\ &\{\{1,a,1\},\{1,a,2\},\{1,b,2\},\{2,a,3\},\{2,a,1\},\{2,b,3\},\{3,a,2\},\{3,b,3\}\},\\ &1,\{1,2\}\} \end{aligned}$$

el conjunt $\{1, 3\}$ i el símbol $a$, el mòdul haurà de tornar el conjunt $\{1, 2\}$.

Donat el mateix AFN, el conjunt $\{2, 3\}$ i el símbol $b$, el mòdul haurà de tornar el conjunt $\{3\}$.

**Exercici 8**

Es demana implementar un mòdul *Mathematica* que, prenent un $AFN$ $A$ i una cadena $x$ com entrada, torne *True* si la cadena és acceptada per l'autòmat i *False* en cas contrari.

**Nota:** Es recomana l'ús de l'exercici 7.

**Exercici 9**

Un autòmat finit determinista es diu que compleix la propietat $P$ si per a tot símbol de l'alfabet, es compleix que les transicions etiquetades amb aquest símbol arriben a un únic estat.

Es demana implementar un mòdul *Mathematica* que, donat un autòmat finit determinista com entrada, torne *True* si l'autòmat cumpleix la propietat i *False* en cas contrari.

**Exemple:**

L'autòmat

$$\begin{aligned} A = \{&\{1, 2, 3\}, \{a, b\},\\ &\{\{1, a, 2\}, \{1, b, 3\}, \{2, a, 2\}, \{2, b, 3\}, \{3, a, 2\}, \{3, b, 3\}\},\\ &1, \{2\}\} \end{aligned}$$

compleix la propietat $P$ mentre que l'autòmat:

$$\begin{aligned} A = \{&\{1, 2, 3, 4\}, \{a, b\},\\ &\{\{1, a, 2\}, \{1, b, 4\}, \{2, a, 4\}, \{2, b, 3\}, \{3, a, 4\}, \{3, b, 2\}, \{4, a, 4\}, \{4, b, 3\}\},\\ &1, \{2\}\} \end{aligned}$$

no la compleix (per exemple, els estats 2 i 4 reben transicions etiquetades amb el símbol $a$).

**Exercici 10**

Donat un autòmat finit determinista complet $A$ i una cadena $u$, es diu que $u$ representa l'autòmat $A$ si a tots els estats de l'autòmat, inclòs l'estat inicial, s'arriba des d'algun altre quan s'analitza la cadena $u$.

Es demana implementar un mòdul *Mathematica* que, donat un autòmat finit determinista i una cadena com entrada, torne devuelva *True* o *False* en funció de que la cadena represente l'autòmat o no.

**Exemple:**

Donat l'autòmat

$$\begin{aligned} A = \{&\{1, 2, 3, 4\}, \{a, b\},\\ &\{\{1, a, 2\}, \{1, b, 1\}, \{2, a, 3\}, \{2, b, 2\}, \{3, a, 4\}, \{3, b, 3\}, \{4, a, 1\}, \{4, b, 4\}\},\\ &1, \{2\}\} \end{aligned}$$

la cadena $u = bba$ representa l'autòmat perquè, en aquest cas, $\delta(1, u) = 2$, $\delta(2, u) = 3$, $\delta(3, u) = 4$ i $\delta(4, u) = 1$.

**Exercici 11**

Donat un autòmat finit determinista complet $A$ i una cadena $u$, es diu que $u$ sincronitza l'autòmat $A$ si l'anàlisi de $u$ des de cada estat de l'autòmat torna el mateix estat de l'autòmat (siga aquest final o no).

Es demana implementar un mòdul *Mathematica* que, donat un autòmat finit determinista i una cadena com entrada, torne devuelva *True* o *False* en funció de que la cadena sincronitze l'autòmat o no.

**Exemple:**

Donat l'autòmat

$$\begin{aligned} A = \{&\{1,2,3,4\}, \{a,b\}, \\ &\{\{1,a,2\}, \{1,b,2\}, \{2,a,2\}, \{2,b,3\}, \{3,a,3\}, \{3,b,4\}, \{4,a,4\}, \{4,b,1\}\}, \\ &1, \{1\}\} \end{aligned}$$

la cadena $u = abbbabbba$ sincronitza l'autòmat perquè, per a tot estat $q$ de l'autòmat, $\delta(q, u) = 2$.