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.

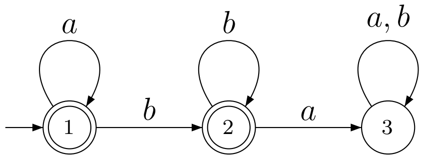
La representació d'Autòmats Finits

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* Σ.
- $trans$ una llista de transicions que representarà la *funció de transició* δ.

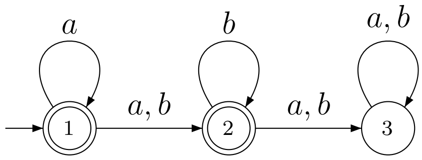
  Cada transició és una llista de tres components $\{EstOrigen, Simbol\ o\ \{\}, EstDestinacio\}$.
- ini es el nom de l'*estad inicial* q_0.
- fin una llista amb el *conjunt d'estats finals* F.

Exemples



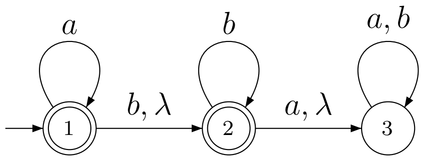
En *Mathematica*:

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



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

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 3

Es demana implementar un mòdul Mathematica que, prenent un AFD A com entrada, torne com eixida un AFD A' sense estats d'absorció (si els tinguera) equivalent a A.

Exercici 4

Donat un autòmat (determinista o no) direm que és codeterminista 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 autòmat A, torne *True* si A és codeterminista i *False* en cas contrari.

Exercici 5

Es demana implementar un mòdul Mathemática 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 6

Es demana implementar un mòdul Mathematica que, donat com entrada un AFD complet $A = (Q, \Sigma, \delta, q_0, F)$ i una cadena $x \in \Sigma^*$, torne $M = (Q, \Sigma, \delta, q_0, F_M)$ amb $F_M = \{q \in Q : \delta(q, x) \in F\}$.

Exercici 7

Es demana implementar un mòdul Mathemática que, prenent com entrada un AFD complet A i una cadena x, torne l'estat de A pel que es passa més vegades durant l'anàlisi de la cadena x.

Exercici 8

Es demana implementar un mòdul Mathemática que, prenent com entrada un AFD complet A i dues cadenes x i y, torne els estats de A que es visiten durant l'anàlisi de les dues cadenes.

Exercici 9

Es demana implementar un mòdul Mathematica que, prenent com entrada un AFN $A = (Q, \Sigma, \delta, q_0, F)$, proporcione com eixida un AFN $A' = (Q, \Sigma, \delta, q_0, F')$ de forma que $F' = \{q \in Q : \forall a \in \Sigma, |\delta(q, a)| \leq 1\}$. (Observeu que els nous estats finals són els estats deterministes de l'autòmat)

Exercici 10

Siga $A = (Q, \Sigma, \delta, q_0, F)$ un AFN. Direm que un estat $q \in Q$ representa el seu alfabet d'entrada si existeixen transicions amb cadascun dels símbols de Σ que eixen de q. Es demana implementar un mòdul Mathematica que, donat com entrada un AFN A, torne *True* si A conté almenys un estat que represente el seu alfabet d'entrada i *False* en cas contrari.

Exercici 11

Es demana implementar un mòdul Mathemática que, prenent com entrada un AFD complet A i dos estats p i q de l'autòmat, torne l'autòmat resultat de substituir qualsevol referència a l'estat q (bé en el conjunt d'estats, el conjunt d'estats finals o les transicions de l'autòmat) per una referència a l'estat p.

Exercici 12

Es demana implementar un mòdul Mathematica que, prenent un $AF - \lambda \quad A$ i un estat q com entrada, torne la $\lambda - clausura$ d'aquest estat.