Introducción

El autómata finito es un modelo formal de un sistema que trabaja con entradas y salidas discretas. El sistema puede estar en cualquiera de las configuraciones de un conjunto finito de configuraciones internas o *estados*. El estado del sistema resume la información concerniente a las anteriores entradas necesaria para determinar el comportamiento del sistema en las subsiguientes entradas.

Aparte del interés teórico del estudio de los autómatas finitos, ya que modelizan una de las familias de lenguajes de la jerarquía de Chomsky, la familia de los *Lenguajes Regulares*, también hay importantes razones prácticas para su estudio, debido a su aplicabilidad al diseño de ciertos tipos de algoritmos muy usados en informática. Por ejemplo, la fase de *análisis léxico* de un compilador está a menudo basada en la simulación de un autómata finito. El analizador léxico analiza los símbolos de un programa fuente para localizar las cadenas de caracteres que corresponden a identificadores, constantes numéricas, palabras reservadas, etc. En este proceso el analizador léxico sólo necesita recordar una cantidad finita de información.

La teoría de autómatas finitos es también muy usada en el diseño de eficientes procesadores de cadenas. Un ejemplo sencillo consiste en el uso de un autómata finito para resolver el problema de detectar en qué posiciones aparece una o un conjunto de distintas palabras (usualmente denominadas patrones) en una palabra más larga x (texto). Este problema se conoce como *String Matching* o *Pattern Matching*.

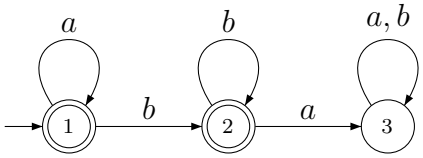
En esta práctica vamos a estudiar la representación de los tres tipos de autómata finito, y la implementación del análisis de palabras sobre los tres tipos de autómatas.

La representación de Autómatas Finitos

Representaremos un Autómata Finito $A = (Q, \Sigma, \delta, q_0, F)$ como una lista $aut = \{est, alf, trans, ini, fin\}$ con cinco componentes que son:

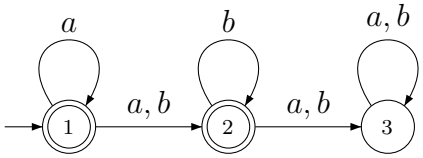
- *est* una lista con los estados del *conjunto de estados* Q (enteros, símbolos, listas, etc.).
- alf una lista con los símbolos del *alfabeto* Σ.
- $trans$ una lista de transiciones que representará la *función de transición* δ.
  Cada transición es una lista de tres componentes $\{EstOrigen, Simbolo\,o\,\{\}, EstDestino\}$.
- *ini* es el nombre del *estado inicial* q_0.
- fin una lista con el *conjunto de estados finales* F.

Ejemplos



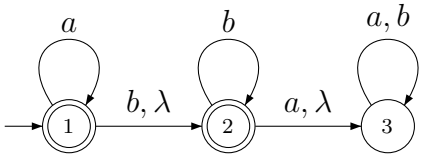
En *Mathematica*:

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



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*

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

Ejercicios

Ejercicio 1

Se pide implementar un módulo Mathematica que, tomando un AF A como entrada, devuelva *True* si A es determinista y *False* en caso contrario.

Ejercicio 2

Dado un AFD, diremos que es bideterminista si tiene un único estado final y no contiene dos transiciones que con el mismo símbolo lleguen al mismo estado. Se pide implementar un módulo Mathematica que, dado como entrada un AFD A, devuelva *True* si A es bideterminista y *False* en caso contrario.

Ejercicio 3

Se pide implementar un módulo Mathematica que, tomando un AFD A como entrada, devuelva *True* si A está completamente especificado y *False* en caso contrario.

Ejercicio 4

Se pide implementar un módulo Mathemática que, tomando un AFD A y una palabra x como entrada, devuelva True si la palabra es aceptada por el autómata y False en caso contrario.

Ejercicio 5

Se pide implementar un módulo Mathematica que, tomando dos AFDs A_1 y A_2, y una palabra x, devuelva si x pertenece al lenguaje $L(A_1) \cup L(A_2)$.

Ejercicio 6

Se pide implementar un módulo Mathematica que, tomando dos AFDs A_1 y A_2, y una palabra x, devuelva si x pertenece al lenguaje $L(A_1) \cap L(A_2)$.

Ejercicio 7

Se pide implementar un módulo Mathemática que, tomando un AFN $A = (Q, \Sigma, \delta, q_0, F)$, un conjunto $C \subseteq Q$ y un símbolo $a \in \Sigma$ como entrada, devuelva $\delta(C, a)$ (el conjunto de estados resultado de analizar en el autómata A el símbolo a a partir de los estados en C).

Ejemplo: Dados el AFN:

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

el conjunto $\{1, 3\}$ y el simbolo a, el módulo deberá devolver el conjunto $\{1, 2\}$.

Dado el mismo AFN, el conjunto $\{2, 3\}$ y el símbolo b, el módulo deberá devolver el conjunto $\{3\}$.

Ejercicio 8

Se pide implementar un módulo Mathemática que, tomando un AFN A y una palabra x como entrada, devuelva True si la palabra es aceptada por el autómata y False en caso contrario.

Nota: Se recomienda el uso del ejercicio 7.

Ejercicio 9

Se pide implementar un módulo Mathematica que, tomando un $AF - \lambda \quad A$ y un estado del mismo q como entrada, devuelva la $\lambda - clausura$ de ese estado.

Ejercicio 10

Un autómata finito determinista se dice que cumple la propiedad P si para todo símbolo del alfabeto, se cumple que las transiciones etiquetadas con ese símbolo alcanzan un único estado.

Se pide implementar un módulo Mathematica que, dado un autómata finito determinista como entrada, devuelva si el autómata cumple la mencionada propiedad.

Ejemplo:

El autómata

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

cumple la propiedad P mientras que el autómata:

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

no la cumple (por ejemplo, los estados 2 y 4 reciben transiciones etiquetadas con el símbolo a).

Ejercicio 11

Dado un autómata finito determinista completo A y una palabra u, se dice que u representa el autómata A si todos los estados del autómata, incluído el estado inicial, son alcanzados desde algún otro cuando se analiza la palabra u.

Se pide implementar un módulo Mathematica que, dado un autómata finito determinista y una palabra como entrada, devuelva True o False en función de que la palabra represente el autómata o no lo haga.

Ejemplo:

Dado el autómata

$$A = \{\{1,2,3,4\},\{a,b\},\\ \{\{1,a,2\},\{1,b,1\},\{2,a,3\},\{2,b,2\},\{3,a,4\},\{3,b,3\},\{4,a,1\},\{4,b,4\}\},\\ 1,\{2\}\}$$

la palabra $u = bba$ representa el autómata porque, en este caso, $\delta(1,u) = 2$, $\delta(2,u) = 3$, $\delta(3,u) = 4$ y $\delta(4,u) = 1$.

Ejercicio 12

Dado un autómata finito determinista completo A y una palabra u, se dice que u sincroniza el autómata A si el análisis de u desde cada estado del autómata devuelve el mismo estado del autómata (sea este final o no).

Se pide implementar un módulo Mathematica que, dado un autómata finito determinista y una palabra como entrada, devuelva True o False en función de que la palabra sincronice el autómata o no lo haga.

Ejemplo:

Dado el autómata

$$A = \{\{1,2,3,4\},\{a,b\},\\ \{\{1,a,2\},\{1,b,2\},\{2,a,2\},\{2,b,3\},\{3,a,3\},\{3,b,4\},\{4,a,4\},\{4,b,1\}\},\\ 1,\{1\}\}$$

la palabra $u = abbbabbba$ sincroniza el autómata porque, para todo estado q del autómata, $\delta(q,u) = 2$.