Autómatas finitos

Un autómata es un modelo matemático que simula una máquina, representada en forma de grafo, cuya función es reconocer una sarta si dicha sarta pertenece al lenguaje y rechazar la sarta si ésta no pertenece al lenguaje. Decimos que es Autómata **finito** porque posee una cantidad **finita de estados**. Se lo llama también “aceptor” porque el sistema acepta las sartas (si pertenecen al lenguaje) o las rechaza (para las que no pertenezcan).

Vamos a definirlo formalmente:

Componentes del AFD quíntupla (Q, A, δ, q0, F ) donde

**Q** Conjunto no vacío y finito de elementos llamados “estados”  
**A** Alfabeto de entrada (símbolos de entrada)  
**δ**: Q x A 🡪 Q Función de transición ( arco orientado) que desde un estado **q** con un elemento digamos “**a**” pasa a otro estado q

**q0** ε Q Estado inicial

**F** Conjunto de estados favorables o de aceptación donde F ∈P(Q) Partes de Q

q

q’

a

Ejemplo de la unción **δ: δ** (q,a) 🡪 q’ que se grafica

Y que se lee: estando en el estado “q” leyendo el símbolo “a” se va / se pasa al estado q’

q’’

b

q’’

q

q’

a

Se leyó el simbola “a” de la sarta ( el primer símbolo de la sarta y lo que queda de la sarta, llamada sarta remanente, la designamos con la letra β

Lon puntos ...... significa que allá habrá tal vez más estados.

Cómo funciona?

El sistema va a procesar cada elemento de la sarta, uno por uno. A medida que vaya leyendo los elementos, va a ir aplicando la función de transición, y va a pasar por los distintos estados del autómata. Si al finalizar de procesar la sarta, ésta pertenece al lenguaje, el autómata se habrá detenido en un estado favorable (o de aceptación). De no pertenecer, la misma va a terminar siendo rechazada.

Veámoslo con un ejemplo. Vamos a generar un autómata que reconozca las sartas de bits cuya cantidad de “1” sea impar.

ej:  
100111 No pertenece al lenguaje  
101001 Pertenece al lenguaje   
001 Pertenece al lenguaje  
1111 No pertenece al lenguaje

Vamos a comenzar definiendo un estado inicial, que vamos a llamar “A”:

A

En nuestro caso, el estado inicial no es de aceptación, porque la sarta λ no tiene una cantidad impar de “1”.

0

A

Nuestro autómata va a ir leyendo elementos de la sarta. Los posibles elementos son “1” ó “0”. En el caso de procesar un 0, el estado no va a cambiar, sigue siendo de rechazo, porque no tiene cantidad impar de 1. Nuestro autómata queda ahora con un bucle en el estado A que indica leer uno o más “0” quedo en A

En el caso de procesar un 1, ya tendríamos un nuevo estado que se indica con un arco orientado del estado A al estado llamado B , es un nuevo estado porque tenemos una cantidad impar de 1. Nuestro autómata ahora es:

1

0

A

B

Nótese que este nuevo estado B está representado por un círculo doble, indicando que es un estado favorable o de aceptación. En este nuevo estado tenemos otra vez la posibilidad de procesar 1 ó 0. En el caso de procesar 0, al igual que en el caso anterior, el autómata se mantiene con cantidad impar de 1, con lo cual su estado sigue siendo de aceptación. Agregamos la transición para el 0:

0

1

0

A

B

Ahora nuevamente, desde el estado B, si procesamos un 1 estaríamos en una cantidad par de 1, lo cual no debería ser aceptado. En este caso vamos a volver al estado anterior A.

0

1

0

A

B

1

Así quedaría el autómata finito desterminista (AFD) para el lenguaje de sarta de bits que tiene sólo cantidad impares de 1.

Para validarlo, vamos a procesar algunas de las sartas que dimos como ejemplo al principio. Para hacerlo vamos a seguir una sencilla tabla. En la primera fila está la sarta como se procesa:

|  |  |  |
| --- | --- | --- |
| **Elemento procesado** | **Estado actual** | **Sarta remanente** |
| - | A | 100111 |
| 1 | B | 00111 |
| 0 | B | 0111 |
| 0 | B | 111 |
| 1 | A | 11 |
| 1 | B | 1 |
| 1 | A | - |

La sarta a probar es 100111 que sabemos que no pertencece al lenguaje ( no tiene una cantidad impar de 1)

En “estado actual” se ve cuál es el estado al que se transiciona después de procesar el elemento de “elemento procesado”.

Como era esperado, la sarta es rechazada (por no terminar en un estado de aceptación).

Si 100111∉L entonces debíamos esperar que 100111 no sea aceptada por el AFD y así fue pues hemos quedado en el estado A ( estado de no aceptacion)

Veamos ahora cómo sería para una sarta que pertenece al lenguaje. Sarta 101001

|  |  |  |
| --- | --- | --- |
| **Elemento procesado** | **Estado actual** | **Sarta remanente** |
| - | A | 101001 |
| 1 | B | 01001 |
| 0 | B | 1001 |
| 1 | A | 001 |
| 0 | A | 01 |
| 0 | A | 1 |
|  | B | - |

En este caso pudidmos terminar de procesar la sarta en el estado B, como era esperado.

101001∈L y por lo tanto debia ser aceptada por el AFD

Veamos ahora un ejemplo más complejo:

(Ejercicio 4.e de la guía):  
Palabras que empiecen con baa V={a,b}

Nuevamente comenzamos con un estado inicial que no es de aceptación, porque la sarta λ no empieza con “baa”. Nuestras opciones con una “b” o con una “a”.

1

a

b

1

3

2

Si comenzamos con un a, sabemos que la sarta no va a poder comenzar con “baa”, así que lo pasamos a otro estado 2.

Si comienza con b, todavía puede ser una sarta aceptada, así que pasamos a un estado 3.

a

b

a

b

a,b

1

3

4

2

Ahora vamos a analizar los estados 2 y 3

Desde el estado 2 ya nunca va a poder comenzar con “baa”, con lo cual cualquier elemento procesado va a quedarse en el mismo estado. Agregamos el bucle “a,b” en el estado.

Desde el estado 3 con una a podemos seguir armando “baa”, y pasamos a un estado nuevo 4. En cambio, con una b estamos en una situación similar a la anterior y pasamos al estado 2.

Finalmente, desde el estado 4 tenemos una situación similar al estado 3. Con una “a” formamos “baa” y pasamos a un nuevo estado 5, que además es de aceptación( ya ahí nuestra sarta o palabra tiene como comienso “baa” que es lo que se pedía) A partir de ahora, no importa qué elemento venga, siempre va a ser aceptada, por eso agregamos el bucle con “a,b” en el estado 5.

Por otro lado, si desde el estado 4 procesamos un b, volvemos a la situación de no poder aceptarla y vamos al estado 2.

a

a

b

a

b

b

a,b

1

3

a,b

4

5

2

Así nos queda dibujado nuestro AFD COMPLETO **(** se llama **COMPLETO** porque desde cada estado está

indicdado a donde voy ( arco rientado) o me quedo( bucle) con cada/todo símbolo de mi vovabulaio,en nuestro caso en cada estado está indicado que se hace al procesar una “a” o una “b” **)**

Nótese que, como mencionamos antes, el estado 2 es un estado en el cual, una vez que llego, no tengo salida es decir que siga como siga la sarta ya no pertenece al lenguaje y no deberá ser aceptada por el Autómata Finito Este estado es conocido como estado “trampa” o “tacho” o etc. Cuando la sarta procesada llega a dicho estado, significa que no es necesario seguir procesándola, porque no puede ser aceptada nunca. ( ya es seguro que es una sarta que no pertenece al Lenguaje pedidio)

Todas las transiciones que no estén explícitamente definidas en el autómata se presume que son transiciones que terminan en un el estado llamado “ trampa” o “tacho” o etc , con lo cual el autómata que sigue es equivalente con el anterior.

a

a

b

1

3

a,b

4

5

Finalmente, podemos decir que un lenguaje aceptado por un autómata finito es el conjunto de todas las sartas del alfabeto tales que luego de aplicar sucesivamente las funciones de transición correspondientes el autómata finaliza en un estado favorable, o de manera formal:

Definición de lenguaje a partir de AF : }