Universidad
Europea de Madrid

LAUREATE INTERNATIONAL UNIVERSITIES

ANÁLISIS LÉXICO

AUTÓMATAS FINITOS

Índice

Presentación

El objetivo de este tema es conocer qué son los autómatas finitos y entender la problemática que resuelve cada uno de los dos tipos de autómata que hay. Además, veremos cómo se representan, realizando distintos ejemplos de **autómatas finitos deterministas (AFD)** y **autómatas finitos no deterministas (AFND).**

Los contenidos que trataremos en este tema son:

- Qué son los autómatas finitos.
- Cómo es un autómata finito.
- Autómata finito determinista (AFD).
- Ejemplo de AFD.
- Autómata finito no determinista (AFND).
- Conceptos de AFND.
- Diagramas de Thomson.
- Ejemplo de expresión regular convertida en AFND.

¿Qué son los autómatas finitos?

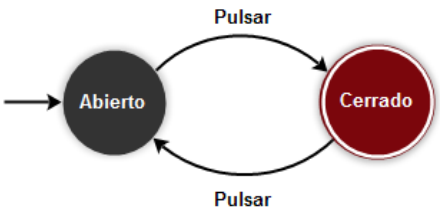
En la década de los 30, A. M. Turing describió una máquina abstracta con las mismas capacidades que los ordenadores que hoy conocemos.

> En las décadas de 1940 y 1950, otros investigadores, entre los que se encontraba C. E. Shannon, estudiaron otros tipos de máquinas más sencillas a las que denominaron autómatas finitos, **definiéndolas como un dispositivo abstracto que es capaz de recibir información, cambiar de estado y comunicar esa información.**

Es importante destacar el trabajo de McCulloch y Pitts, que intenta modelar el comportamiento de una neurona, constituyendo la célula de McCulloch-Pitts donde definieron una serie de propiedades muy útiles para los autómatas finitos. Por otro lado, a finales de 1950, Noam Chomsky estudió las **gramáticas formales, que tienen una relación muy cercana con los autómatas finitos pese a no ser máquinas abstractas.** De hecho, los autómatas finitos constituyen un modelo para explicar el comportamiento de estas gramáticas formales, sobre todo las que en la jerarquía de Chomsky se corresponden con las de tipo 3.

De lo que se trata con un autómata finito es de poder decir en qué estado está un determinado sistema (en este caso autómata) entre un conjunto finito de estados. Por tanto, el objetivo es conocer, o recordar, los estados por los que se ha ido pasando para llegar al estado actual. Que los diferentes estados en los que puede estar el autómata constituyan un conjunto finito, tiene una gran ventaja desde el punto de vista de la implementación: **vamos a necesitar un conjunto finito de recursos.**

El ejemplo típico para explicar el autómata finito más sencillo que existe es el que representa un interruptor. Este sistema es capaz de saber en que estado está: abierto o cerrado, y pasamos de un estado a otro pulsando un botón.



Vemos que hay un estado inicial (por convenio es el que tiene la flecha) que, en nuestro caso, es abierto. Es conveniente también indicar el estado final, o de aceptación, al que llegamos después de varios pasos (o entradas). Por convenio se suele identificar por un doble círculo.

¿Cómo representamos un autómata finito?

Para definir un autómata finito necesitamos cinco elementos:

$$AF = (\Sigma, Q, f, q_0, F)$$

Es el alfabeto de símbolos de entrada.

$$AF = (\Sigma, Q, f, q_0, F)$$

Es el conjunto de estados.

$$AF = (\Sigma, Q, f, q_0, F)$$

Es la función de transición entre estados.

$$AF = (\Sigma, Q, f, q_0, F)$$

Es el estado inicial.

$$AF = (\Sigma, Q, f, q_0, F)$$

Es el conjunto de estados finales.

Un autómata finito se puede representar de dos formas: mediante su tabla o su diagrama de transiciones.

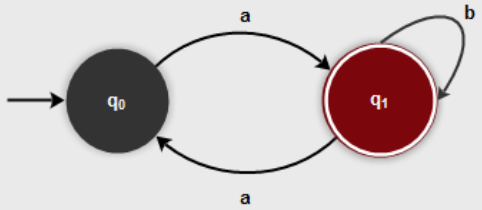
Los autómatas finitos pueden ser de dos tipos: **autómatas finitos deterministas (AFD) y autómatas finitos no deterministas (AFND).**

La diferencia principal entre ellos es que los AFD tienen una sola transición por cada entrada, desde cada estado, mientras que en los AFND puede existir más de una transición por cada entrada, desde cada estado, o bien ninguna transición. Además los AFND pueden hacer transiciones de un estado a otro sin leer ningún símbolo de la entrada (λ), a estas transiciones se les denomina transiciones-λ.

Tanto los AFD como los AFND pueden reconocer expresiones regulares, aunque desde el punto de vista de la implementación, nos interesa utilizar los AFD. El ejemplo que hemos utilizado para el diagrama y la tabla de transiciones es de un autómata finito determinista.

Tabla y diagrama de transiciones

f	a	b
$\rightarrow q_0$	q_1	q_1
$^*q_1$	q0	q_1



Autómata finito determinista (AFD)

Como hemos indicado, para definir formalmente un AFD, lo hacemos de la siguiente forma: **AFD = ($\sum$, Q, f, q_0, F).**

La función de transición (**f**) especifica cómo el autómata pasa de un estado a otro, en función del carácter que recibamos en la entrada. Por tanto si estamos en el estado q_X y llamamos **W** a la palabra que se está reconociendo, la función de transición nos indicará cómo pasamos al estado q_Y al reconocer determinado carácter **a** de esa palabra **W.**

La forma de especificar esta transición utilizando una notación matemática es la siguiente:

$$(q_X, aW) \rightarrow (q_Y, W), \text{ cumpliéndose que } f(q_X, a) = q_Y$$

Un AFD se caracteriza por lo siguiente:

- Para cada estado del conjunto de estados Q, y cada símbolo de la entrada, hay solo una arista etiquetada con ese símbolo de la entrada que sale de ese estado y va a otro.
- No existen las transiciones-λ, es decir no hay transiciones con la cadena vacía.

Como conceptos importantes asociados a los AFD, podemos citar:

Lenguaje reconocido por un AFD	Es el conjunto de palabras que acepta o reconoce el AFD. Por otro lado, cuando el autómata no es capaz de alcanzar un estado final, se dice que esa palabra no pertenece al lenguaje que reconoce el autómata. La manera formal de describir este concepto es la siguiente: $L_{AFD} = \{ x \mid x \in \Sigma^* \wedge F(q0, x) \in F\}$
Equivalencia de AFD	Dos autómatas finitos deterministas son equivalentes si y solo si aceptan el mismo lenguaje. Una de las aplicaciones más interesantes de este concepto, es que siempre existirá un **AFD mínimo equivalente para cada AFD.**

AF asociado a gramáticas de tipo 3	Para todo autómata finito (no siempre será AFD) existe una gramática regular (de tipo 3), de tal forma que el lenguaje que genera esta gramática es igual al lenguaje reconocido por el autómata finito: **$L(G_3) = L(AF)$**

Ejemplo de AFD

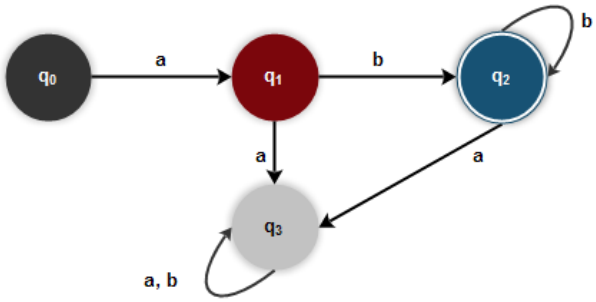
A partir del **AFD = (∑, Q, f, q_0, F)**, donde **∑ = {a,b}, Q = {q_0, q_1, q_2, q_3}**, la función de transición f está determinada por la siguiente tabla y el conjunto de estados finales **F = {q_2}.**

f	a	b
q_0	q_1	-
q_1	q_3	q_2
q_2	q_3	q_2
q_3	q_3	q_3

Vamos a identificar el lenguaje que reconoce este autómata, además de la expresión regular equivalente:

Paso 1

Realizar el diagrama de transición.



Paso 2

Identificar el lenguaje que reconoce.

$$L(\sum_1) = \{ab, abb, abbb, abbbb, ...\}$$

Paso 3

Obtener la expresión regular equivalente: **ab^+**,puesto que siempre habrá al menos una b. Otra forma de decir lo mismo es **abb*.**

Autómata finito no determinista (AFND)

En este tipo de autómatas puede existir más de una transición para cada estado y sus posibles entradas, o bien ninguna transición.

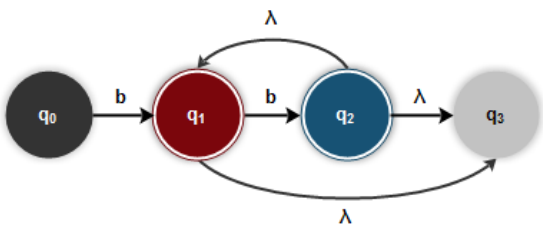
Otra característica que los diferencia de los AFD, son las **transiciones-λ**, es decir, transiciones de un estado a otro sin que se haya leído ningún símbolo de la entrada. La manera formal de definir un AFND, es idéntica a la de definir un AFD, excepto por la función de transición que se utiliza y que los hace diferentes:

$$\mathbf{AFND = (\sum, Q, f, q_0, F)}$$

Todo es igual excepto que, para **f,** incluye las transiciones-λ, es decir, que para una misma entrada se puede transitar a más de un estado. Esto se refleja tanto en la tabla como en el diagrama de transiciones.

f	b	λ
→q_0	q_1	∅
*q_1	q_2	q_3
*q_2	∅	q_3, q_1
*q_3	∅	∅

Paso 1. Realizar el diagrama de transición →



Paso 2. Identificar el lenguaje que reconoce:

$$\mathbf{L(\sum_1) = \{b, bb, bbb, bbbb, ...\}}$$

Paso 3. Obtener la expresión regular equivalente: **b+** puesto que siempre habrá al menos una b. Otra forma de decir lo mismo es **bb*.**

Conceptos del AFND

Lenguaje asociado a un AFND: es el conjunto de palabras que le hacen transitar desde el estado inicial a algún estado final, utilizando para ello la función **f.**

Equivalencia entre AFD y AFND: para cada AFD existe un AFND equivalente y viceversa. De hecho los AFD son un caso particular de los AFND. En la práctica, el AFD tiene casi el mismo número de estados que el AFND, aunque normalmente tiene más transiciones.

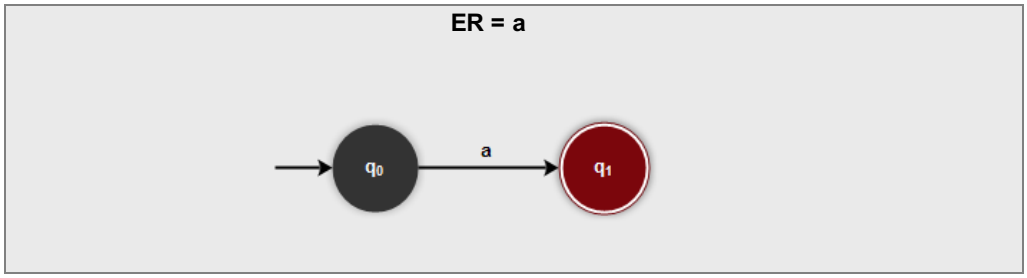
Transformación de AFND en AFD: de todo AFND se puede obtener un AFD de tal forma que el lenguaje que reconozca el AFD sea equivalente al lenguaje reconocido por el AFND.

Transformación de una expresión regular en un autómata finito: a partir de una expresión regular se puede obtener el autómata finito (determinista o no) capaz de reconocer el lenguaje que dicha expresión representa y viceversa, es decir, a partir de un autómata finito se puede obtener la expresión regular que este autómata representa.

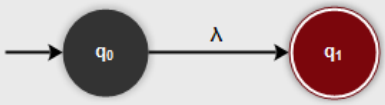
Para poder realizar esta transformación, necesitamos conocer las equivalencias entre las ER básicas y los autómatas finitos que las representan.

Equivalencias entre expresiones regulares básicas y autómatas finitos: vamos a ver las expresiones regulares básicas: repetición en sus distintas variantes, alternativa y concatenación y su equivalente AF.

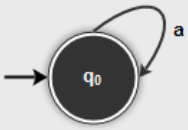
Equivalencias
ER = a
ER = λ
ER = a* (Cero o más repeticiones de a)
ER = a+ = aa* (Una o más repeticiones de a)
ER = a|b (Alternativa)
ER = a? = a|λ (Cero o una instancia de a)
ER = ab (Concatenación)



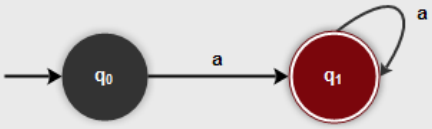
ER = λ



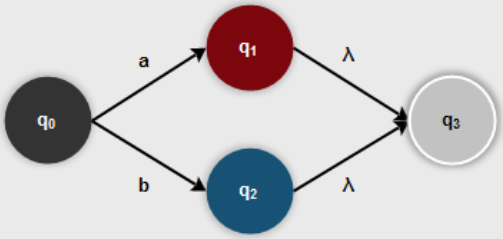
ER = a* (cero o más repeticiones de a)



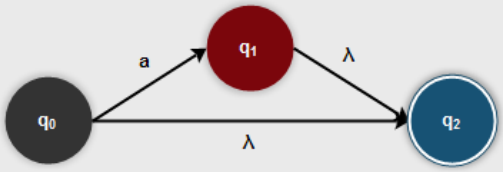
ER = a+ = aa* (una o más repeticiones de a)



ER = a|b (Alternativa)



ER = a? = a|λ (cero o una instancia de a)



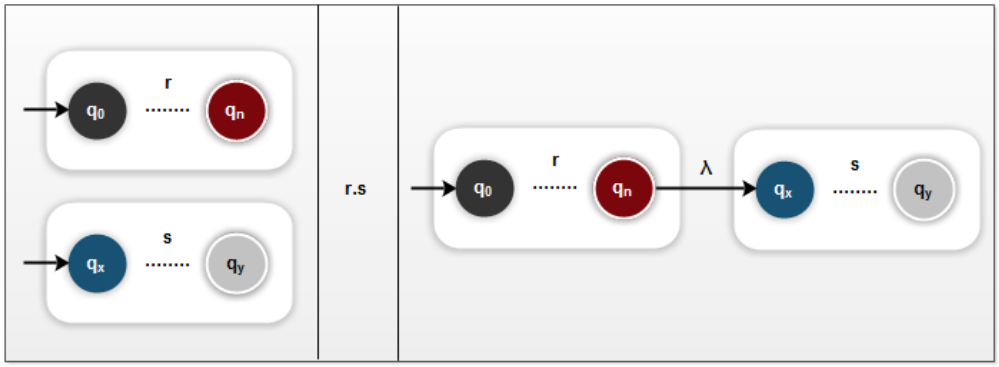
ER = ab (Concatenación)

Diagramas de Thomson

A partir de los diagramas básicos anteriores podemos obtener el resto de diagramas de transición para un AFND, pero para representar estos diagramas de transiciones tenemos que tener en cuenta todas las posibilidades y la que nos afecta aquí son las transiciones-λ que generan transiciones sin leer ningún carácter de la entrada.

Los diagramas o construcciones de Thomson, en honor a su inventor (Louden, 2004), utilizan transiciones-λ para, enlazando los autómatas de cada segmento de una expresión regular, construir el autómata completo. Partiendo de las expresiones regulares básicas que acabamos de ver, realizaremos las construcciones de Thomson para cada tipo de operación.

Concatenación: supongamos dos segmentos r y s cada uno de ellos con sus distintas operaciones básicas, su concatenación sería:



Documentos Operaciones alternativa y repetición

Transiciones-λ

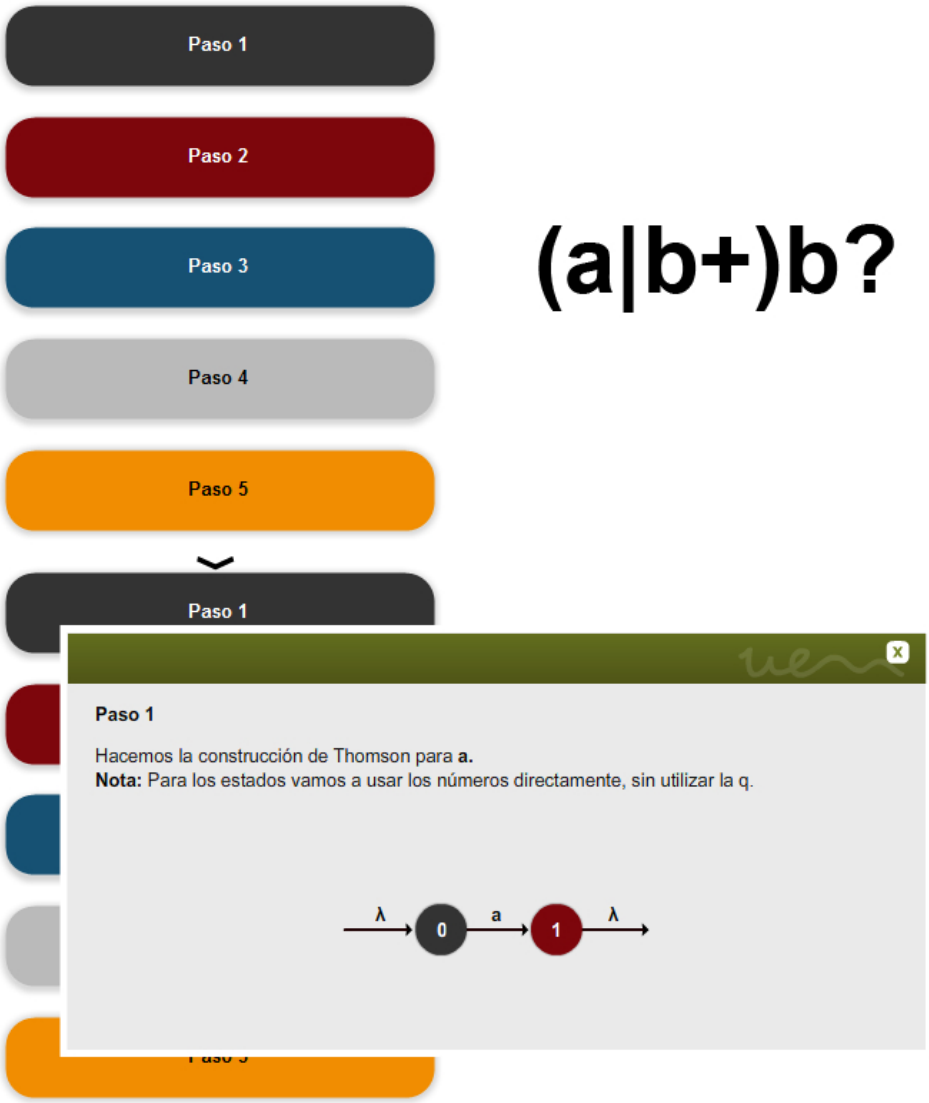
Transiciones-λ que generan transiciones sin leer ningún carácter de la entrada

Al tratar estas transiciones debemos tener en cuenta que en ellas se incluyen las transiciones-λ que se puedan producir antes de leerse el primer símbolo de la entrada, y también las que pueda haber después de leerse el último símbolo de la entrada.

Ejemplo de ER convertida en AFND

A partir de una ER y de las construcciones de Thomson vamos a obtener el AFND equivalente.

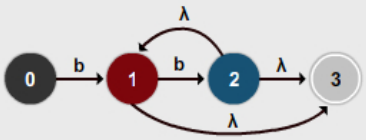
En este caso la ER será: **(a|b+)b?.**



Paso 1

Paso 2

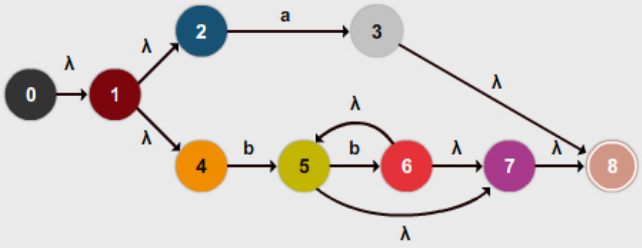
Obtenemos la construcción de Thomson de **b+** (recordemos que es igual que bb*).



Paso 1

Paso 3

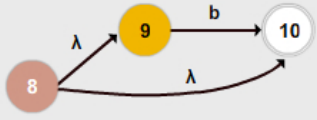
Realizamos la alternativa **a|b+.**

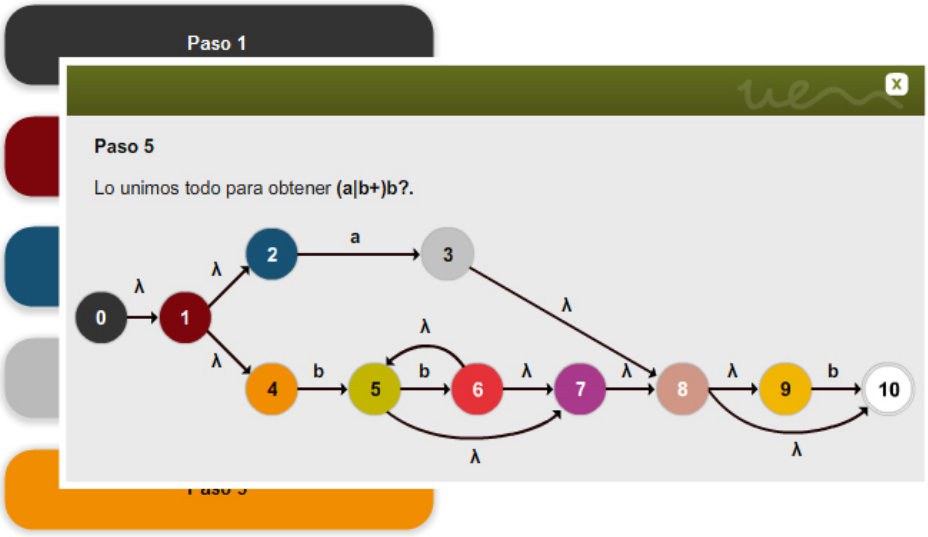


Paso 1

Paso 4

Realizamos **b?**, que indica cero o una instancia de b.





Resumen

En este tema hemos entendido qué son, para qué sirven y cómo se representan los dos tipos existentes de autómatas finitos: autómatas finitos deterministas (AFD) y autómatas finitos no deterministas (AFND).

En primer lugar hemos visto cómo se caracterizan los AFD, y qué lenguaje reconocen. Además hemos visto un ejemplo de AFD representado por una tabla y un diagrama de transiciones.

En segundo lugar, hemos estudiado los AFND, que se caracterizan por las transiciones-λ y porque desde un estado y con una sola entrada podemos transitar a más de un estado. También hemos visto ejemplos de ellos y tratado conceptos necesarios como el lenguaje que reconocen, la equivalencia entre AFD y AFND y por último la transformación de una expresión regular (ER) en un autómata finito (AF), tratando las ER básicas y su equivalente entre los AF.

Hemos entendido la utilidad de los diagramas o construcciones de Thomson y su utilidad para representar AFND y cómo, utilizando las transiciones-λ, enlazamos las distintas partes de una ER y así obtenemos el autómata completo.

Para finalizar hemos realizado un ejemplo completo de paso de ER a AFND aplicando los conceptos aprendidos en las construcciones de Thomson.