Autómatas finitos deterministas

Nuestro problema principal es determinar si una palabra pertenece ó no a un lenguaje. Por ejemplo, si A es el lenguaje de la expresión regular $c^*(a \cup bc^*)^*$ entonces

$$abc^5c^3ab \in A, \quad cabac^3bc \notin A,$$

el análisis se puede hacer letra por letra según sus posiciones. Para ayudar a tal anásisis se hace uso de *grafos dirigidos* llamados *diagramas de transición*. Los nodos de tales grafos se llaman *estados*, las flechas se llaman *transiciones* y se etiquetan éstas flechas con símbolos del alfabeto.
Hay símbolos especiales: *estado inicial* que se marca con $\rightarrow$ y *estados finales* ó de *aceptación* que se marcan con un círculo: $\odot$

Por ejemplo:

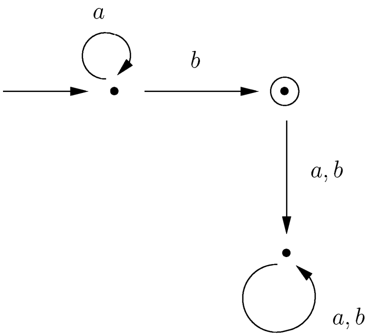
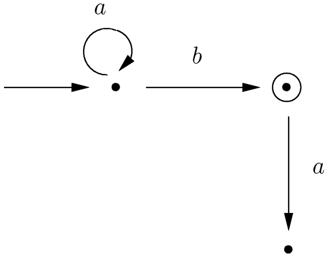


Figure: 2. Un autómata finito determinista

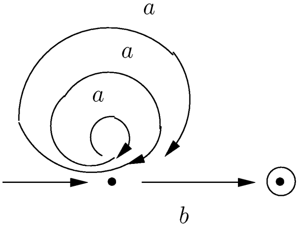
Tales grafos se llaman *autómatas finitos deterministas* (AFD).

Una palabra se dice *aceptada* ó *legal* con respecto a un AFD si partiendo del estado marcado como inicial, se llega a un estado de aceptación mediante el siguiente procedimiento:
¿la cadena *aba* es aceptada por el AFD de la figura 2? comenzando del nodo marcado como estada inicial seguimos el camino indicado por las flechas con etiquetas las letras de la palabra en cuestión:



se arriba entonces a un estado que no es de aceptación, por lo que la palabra *aba* se rechaza.

¿$a^3 b$ es aceptada? veamos el diagrama:



como se puede notar, tal palabra nos hace llegar al estado de aceptación, por lo que la palabra $a^3 b$ es aceptada.
De donde es claro que el autómata finito determinista de la figura 2 acepta las palabras del lenguaje de $a^* b$.

Ejemplo

Sea $\Sigma = \{a, b\}$. Consideremos el lenguaje

$$A = \{(ab)^i \mid i \geq 1\} \ .$$

Construir un AFD que acepte únicamente a las palabras de A.
Sol. Recordemos que

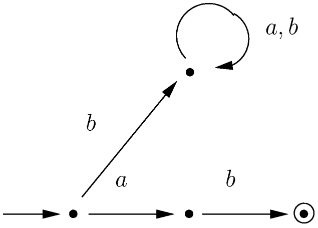
$$A = \{ab, abab, ababab, \ldots\}$$

de donde al menos la palabra ab debe, en el autómata que construyamos, conducir a un estado de aceptación. Lo que sugiere que consideremos el diagrama

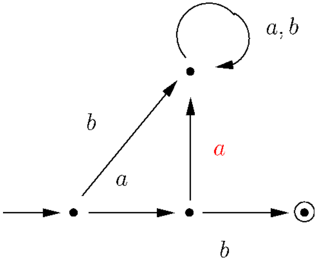
$$\longrightarrow \bullet \xrightarrow{\quad a \quad} \bullet \xrightarrow{\quad b \quad} \odot$$

éste aún no es un AFD, puesto que se requiere que el autómata, en cada estado, sepa a qué estado nuevo se transita ante la aparición de cualquier letra del alfabeto, en nuestro caso a y b.

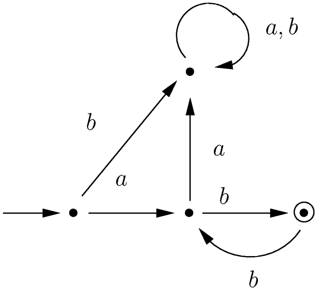
Notemos que en nuestro primer diagrama el autómata, en el estado inicial no sabrá que hacer si aparece una b. Ninguna palabra de A tiene prefijo b, luego las palabras que comiencen con b, sin importar lo que siga, deben de ser rechazadas. Esto sugiere:



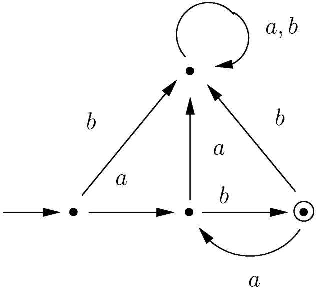
Otras palabras a rechazar son aquellas que después de a, en lugar de continuar con b, continuen con a, lo que sugiere



Otras palabras que deben de ser aceptadas son por ejemplo *abab*, *ababab*. La manera de crear repeticiones es introducir ciclos en el grafo:



con lo cual aceptamos las palabras del tipo $(ab)^{+}$. Pero aún debemos rechazar las palabras que al tener prefijo *ab* continuen con *b*:

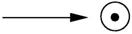


lo que completa nuestro autómata finito determinista que acepta solamente las palabras del lenguaje de $(ab)^+$.

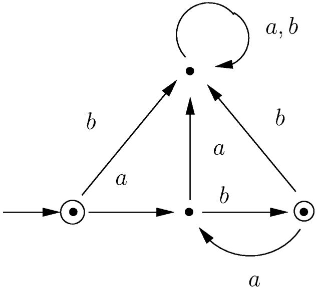
Ejemplo

Lo mismo que el anterior para $A = (ab)^*$.

Sol. Ahora la palabra vacía también tiene que ser aceptada. La técnica para aceptar a ϵ es hacer al estado inicial, final también:

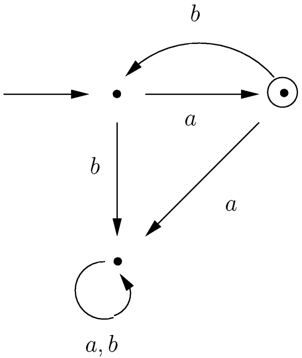


Por lo que el autómata pedido es



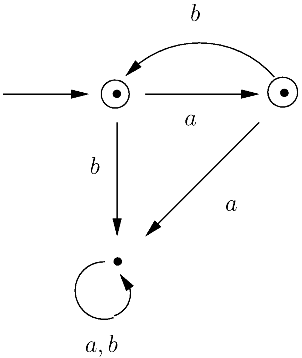
Nótese que ahora tenemos *dos* estados finales.

En general cuando en un AFD se hace del estado inicial un estado final, no sólo se va ha aceptar a la palabra vacía, puede que se acepten otras indeseables. Por ejemplo, en el AFD,



se acepta sólo a al lenguaje $a(ba)^*$ y no a la palabra vacía.

Si ponemos al estado inicial como final obtenemos



que acepta no sólo a la palabra vacía ϵ, sino que se "cuela" todo el lenguaje $(ab)^*$. Es decir, el nuevo autómata acepta a $(ab)^* \cup a(ba)^*$.