2.4. Autómatas finitos no deterministas (AFN)

Los **autómatas finitos no-deterministas** (AFN) se asemejan a los AFD, excepto por el hecho de que para cada estado $q \in Q$ y cada $a \in \Sigma$, la transición $\delta(q, a)$ puede consistir en más de un estado o puede no estar definida. Concretamente, un AFN está definido por $M = (\Sigma, Q, q_0, F, \Delta)$ donde:

1. Σ es el alfabeto de cinta.

2. Q es un conjunto (finito) de estados.

3. $q_0 \in Q$ es el estado inicial.

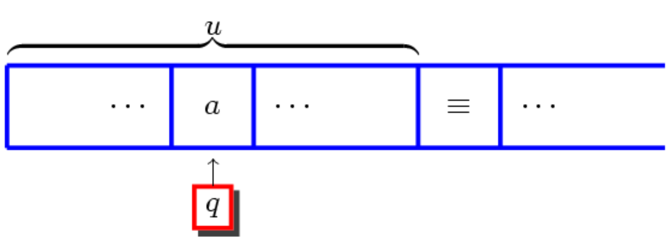
4. $\varnothing \neq F \subseteq Q$ es el conjunto de estados finales o estados de aceptación.

5. $$\begin{aligned} \Delta : Q \times \Sigma &\longrightarrow \wp(Q) \\ (q, a) &\longmapsto \Delta(q, a) = \{q_{i_1}, q_{i_2}, \ldots, q_{i_k}\} \end{aligned}$$

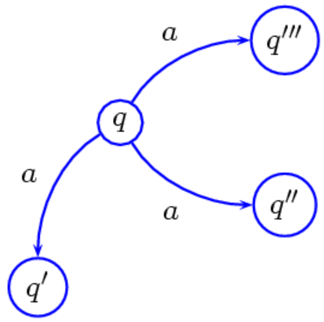
   donde $\wp(Q)$ es el conjunto de subconjunto de Q.

Puede suceder que $\Delta(q, a) = \varnothing$, lo cual significa que, si durante el procesamiento de una cadena de entrada u, M ingresa al estado q leyendo sobre la cinta el símbolo a, el cómputo se aborta.

Cómputo abortado:



La noción de diagrama de estados para un AFN se define de manera análoga al caso AFD, pero puede suceder que desde un mismo nodo (estado) salgan dos o más arcos con la misma etiqueta:



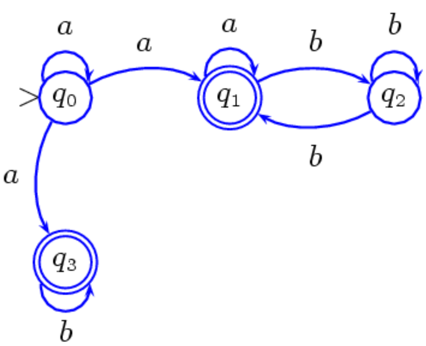
Un AFN M puede procesar una cadena de entrada $u \in \Sigma^*$ de varias maneras. Sobre el diagrama de estados del autómata, esto significa que pueden existir varias trayectorias etiquetadas con los símbolos de u.

La siguiente es la noción de aceptación para autómatas no deterministas:

$$\boxed{\begin{array}{rcl} L(M) & = & \text{lenguaje aceptado o reconocido por } M \\ & = & \{u \in \Sigma^* : \text{existe por lo menos un cómputo completo} \\ & & \qquad\qquad \text{de } u \text{ que termina en un estado } q \in F\} \end{array}}$$

Es decir, para que una cadena u sea aceptada, debe existir por lo menos un cómputo en el que u sea procesada completamente y que finalice estando M en un estado de aceptación.

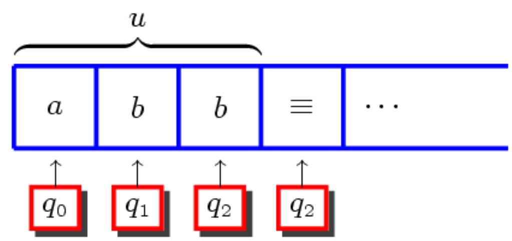
Ejemplo Sea M el siguiente AFN:



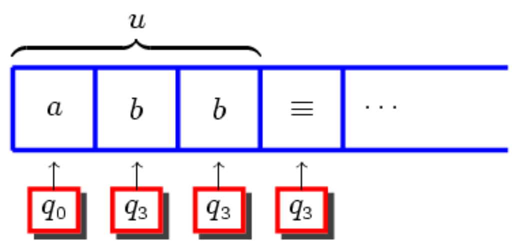
Δ	a	b
q_0	$\{q_0, q_1, q_3\}$	$\varnothing$
q_1	$\{q_1\}$	$\{q_2\}$
q_2	$\varnothing$	$\{q_1, q_2\}$
q_3	$\varnothing$	$\{q_3\}$

Para la cadena de entrada $u = abb$, existen cómputos que conducen al rechazo, cómputos abortados y cómputos que terminan en estados de aceptación. Según la definición de lenguaje aceptado, $u \in L(M)$.

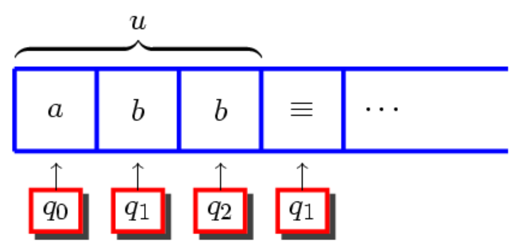
Cómputo de rechazo:



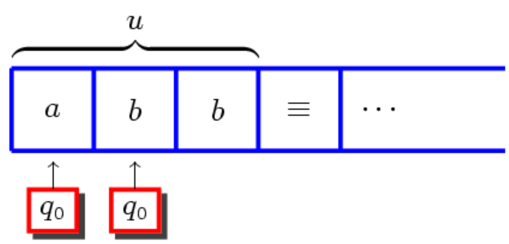
Cómputo de aceptación:



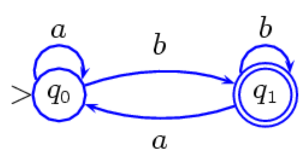
Otro cómputo de aceptación:



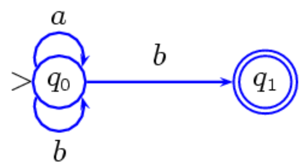
Cómputo abortado:



Ejemplo En el último ejemplo de la sección 2.3 se diseñó el AFD que acepta el lenguaje de las cadenas sobre $\Sigma = \{a, b\}$ que terminan en b:

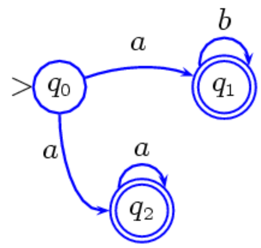


Un AFN que acepta el mismo lenguaje y que es, tal vez, más fácil de concebir, es el siguiente:

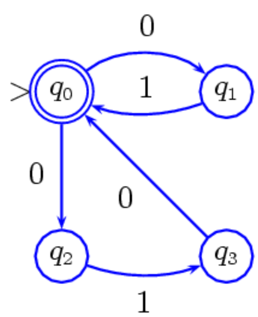


Este autómata se asemeja a la expresión regular $(a \cup b)^*b$.

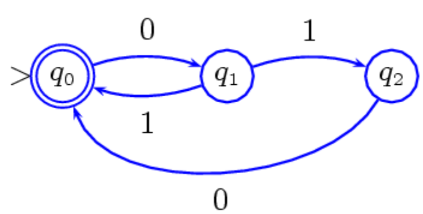
Ejemplo Considérese el lenguaje $L = ab^* \cup a^+$ sobre el alfabeto $\Sigma = \{a, b\}$. El siguiente AFN M satisface $L(M) = L$.



Ejemplo $\Sigma = \{0, 1\}$, $L = (01 \cup 010)^*$. El siguiente AFN acepta a L.



Otro AFN que acepta el mismo lenguaje y que tiene sólo tres estados es el siguiente:



Ejercicios Diseñar autómatas AFD o AFN que acepten los siguientes lenguajes:

1. $\Sigma = \{a, b\}$. $L = ab^+a^*$.
2. $\Sigma = \{a, b\}$. $L = a(a \cup ab)^*$.
3. $\Sigma = \{a, b, c\}$. $L = a^*b^*c^*$.
4. $\Sigma = \{0, 1, 2\}$. $L =$ lenguaje de las cadenas sobre Σ que comienzan con 0 y terminan con 2.
5. $\Sigma = \{0, 1\}$. Lenguaje de las cadenas de longitud par ≥ 2 formadas por ceros y unos alternados.
6. $\Sigma = \{0, 1\}$. Lenguaje de las cadenas que tienen a lo sumo dos ceros consecutivos.
7. $\Sigma = \{0, 1\}$. Lenguaje de las cadenas de longitud impar que tienen unos en todas las posiciones impares y únicamente en las posiciones impares.
8. $\Sigma = \{a, b, c\}$. $L =$ lenguaje de las cadenas sobre Σ que contienen la cadena bc.
9. $\Sigma = \{a, b, c\}$. $L =$ lenguaje de las cadenas sobre Σ que *no* contienen la cadena bc. En el último ejemplo de la sección 1.14 se presentaron dos expresiones regulares para L. Nota: ¡se puede construir un AFD con sólo dos estados para aceptar este lenguaje!