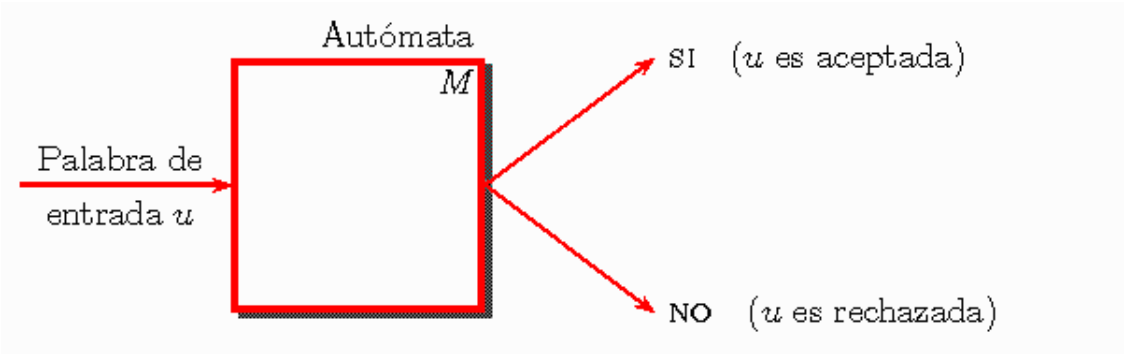
Capítulo 2

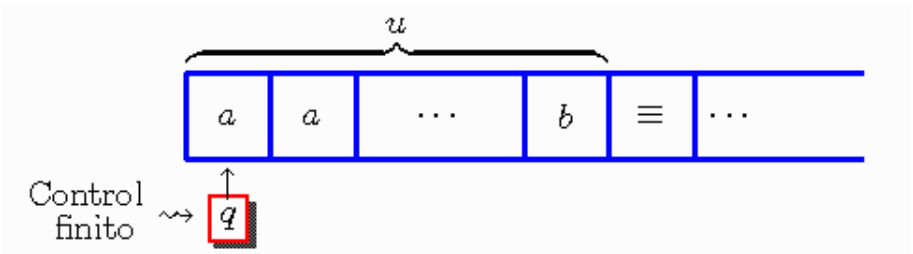
Autómatas finitos

2.1. Autómatas finitos deterministas

Los **autómatas finitos** son máquinas abstractas que procesan palabras, las cuales son aceptadas o rechazadas.



El autómata actúa leyendo los símbolos escritos sobre una cinta semi-infinita, dividida en casillas, sobre la cual se escribe una palabra de entrada u, un símbolo por casilla. El autómata posee una **cabeza lectora** o **control finito**, que inicialmente "escanea" o lee la casilla del extremo izquierdo de la cinta.



La cabeza lectora del autómata posee un cierto número (finito) de configuraciones internas, llamadas **estados del autómata**. Entre los estados de un autómata se destacan el **estado inicial** y los **estados finales** o **estados de aceptación**.

Formalmente, un autómata finito M está definido por cinco parámetros,

$$M = (\Sigma, Q, q_0, F, \delta),$$

a saber:

1. Un alfabeto Σ, llamado alfabeto de cinta. Todas las palabras que procesa M pertenecen a Σ^*.
2. $Q = \{q_0, q_1, \ldots, q_n\}$, conjunto de estados del autómata.
3. $q_0 \in Q$, estado inicial.
4. $F \subseteq Q$, conjunto de estados finales o de aceptación. $F \neq \varnothing$.
5. La función de transición del autómata

$$\begin{array}{rcl} \delta : Q \times \Sigma & \longrightarrow & Q \\ (q, a) & \longmapsto & \delta(q, a) \end{array}$$

Toda palabra de entrada es procesada completamente, hasta que la cabeza lectora encuentra la primera casilla vacía.

Ejemplo

$\Sigma = \{a, b\}$

$Q = \{q_0, q_1, q_2\}$

q_0 : estado inicial

$F = \{q_0, q_2\}$, estados de aceptación.

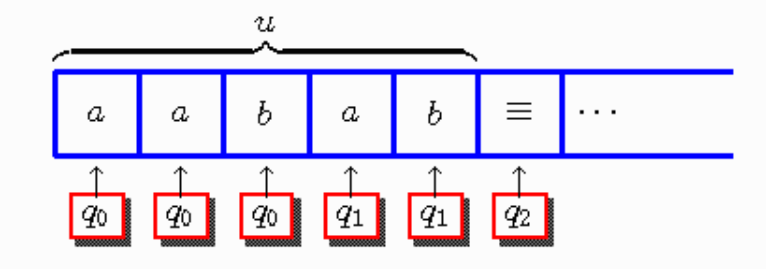
Función de transición δ:

δ	a	b
q_0	q_0	q_1
q_1	q_1	q_2
q_2	q_1	q_1

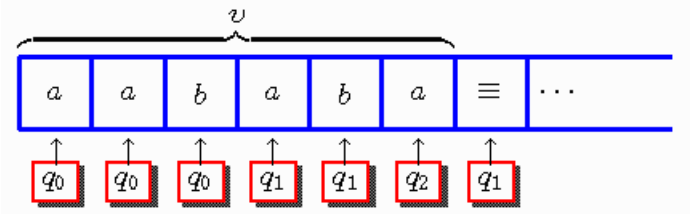
$$\begin{array}{ll} \delta(q_0, a) = q_0 & \delta(q_0, b) = q_1 \\ \delta(q_1, a) = q_1 & \delta(q_1, b) = q_2 \\ \delta(q_2, a) = q_1 & \delta(q_2, b) = q_1 \end{array}$$

Ilustración del procesamiento de dos palabras de entrada:

1. $u = aabab$.
2. $v = aababa$.

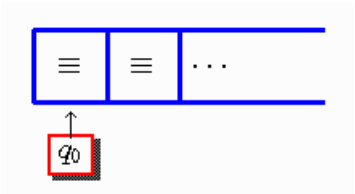


Puesto que q_2 es un estado de aceptación, la palabra de entrada u es aceptada.



Puesto que q_1 no es un estado de aceptación, la palabra de entrada v es rechazada.

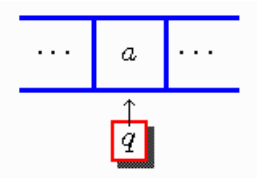
Caso especial: la palabra λ es la palabra de entrada.



Puesto que q_0 es un estado de aceptación, la palabra λ es aceptada.

En general se tiene: la palabra vacía λ es aceptada por un autómata M si y solamente si el estado inicial q_0 de M también es un estado de aceptación.

Los autómatas finitos descritos arriba se denominan **autómatas finitos determinitas** ya que para cada estado q y para cada símbolo $a \in \Sigma$, la función de transición $\delta(q, a)$ siempre está definida. Es decir, la función de transición δ *determina unívocamente* la acción que el autómata realiza cuando el control finito se encuentra en un estado q leyendo un símbolo a sobre la cinta:



Dado un autómata M, el **lenguaje aceptado o reconocido** por M se denota $L(M)$ y se define por

$$L(M) := \{u \in \Sigma^* : M \text{ termina el procesamiento de la palabra de entrada } u \text{ en un estado } q \in F\}.$$

2.2. Diagrama de estados de un autómata finito

Un autómata finito se puede representar por medio de un grafo dirigido y etiquetado. Recuérdese que un **grafo** es un conjunto de vértices o nodos unidos por arcos o conectores; si los arcos tienen tanto dirección como etiquetas, el grafo se denomina **grafo dirigido y etiquetado** o **digrafo etiquetado**.

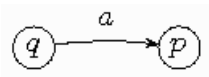
El grafo de un autómata se obtiene siguiendo las siguientes convenciones:

- Los vértices o nodos son los estados del autómata.
- El estado q se representa por: 

- El estado inicial q_0 se representa por: 

- Un estado final q se representa por: 

- La transición $\delta = (q, a) = p$ se representa en la forma:



Dicho grafo se denomina el **diagrama de estados del autómata**.

Ejemplo Diagrama de estados del autómata de la sección anterior.

$\Sigma = \{a, b\}$

$Q = \{q_0, q_1, q_2\}$

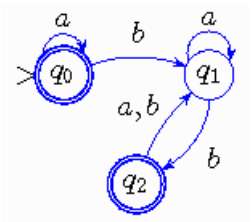
q_0 : estado inicial

$F = \{q_0, q_2\}$, estados de aceptación.

Función de transición δ:

δ	a	b
q_0	q_0	q_1
q_1	q_1	q_2
q_2	q_1	q_1

$$\delta(q_0, a) = q_0 \qquad \delta(q_0, b) = q_1$$
$$\delta(q_1, a) = q_1 \qquad \delta(q_1, b) = q_2$$
$$\delta(q_2, a) = q_1 \qquad \delta(q_2, b) = q_1$$



2.3. Diseño de autómatas

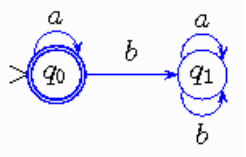
Para autómatas deterministas se adopta la siguiente convención adicional con respecto a los diagramas de estados: se supone los arcos no dibujadas explícitamente conducen a un estado "limbo" de no-aceptación. Es decir, en el diagrama de estados se indican únicamente los arcos que conduzcan a trayectorias de aceptación. Esto permite simplificar considerablemente los diagramas.

En esta sección se considerará el siguiente tipo de problemas:

> Dado un lenguaje regular L diseñar un autómata finito deterministas M que acepte o reconozca a L, es decir, tal que $L(M) = L$.

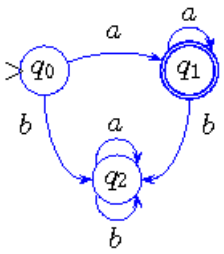
Más adelante se demostrará, en toda su generalidad, que estos problemas *siempre* tienen solución.

Ejemplo $\Sigma = \{a, b\}$. $L = a^* = \{\lambda, a, a^2, a^3, \ldots\}$.

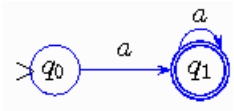


Versión simplificada:

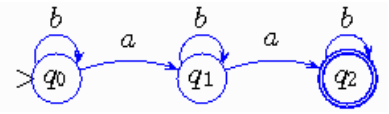
Ejemplo $\Sigma = \{a, b\}$. $L = a^+ = \{a, a^2, a^3, \ldots\}$.



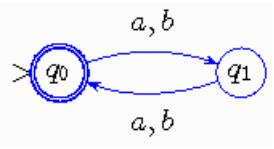
Versión simplificada:



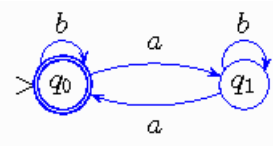
Ejemplo $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ que contienen exactamente dos a's. $L = b^*ab^*ab^*$.



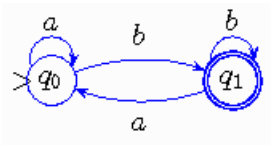
Ejemplo $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ que tienen un número par de símbolos (palabras de longitud par).



Ejemplo $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ que contienen un número par de a's.



Ejemplo $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ que terminan en b.



Ejercicios Diseñar autómatas finitos deterministas que acepten los siguientes lenguajes:

1. $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ de longitud impar.
2. $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ que contienen un número impar de b's.
3. $\Sigma = \{a, b, c\}$. $L =$ lenguaje de las palabras sobre Σ que contienen la cadena bc.
4. $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ que comienzan con b y terminan con a.
5. $\Sigma = \{a, b\}$. $L =$ lenguaje de las palabras sobre Σ que contienen un número par de a's y un número par de b's. Ayuda: utilizar 4 estados.
6. $\Sigma = \{a, b\}$. $L = ab^+$.
7. $\Sigma = \{a, b\}$. $L = ab^* \cup ab^*a$.
8. $\Sigma = \{a, b\}$. $L = (a \cup ba)^*$.
9. $\Sigma = \{a, b\}$. $L = (ab \cup ba)^*$.

2.4. Autómatas finitos no deterministas

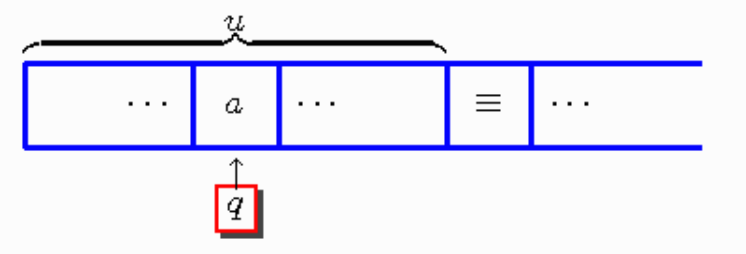
Los autómatas finitos no-deterministas se asemejan a los AFD, excepto por el hecho de que para cada estado $q \in Q$ y cada $a \in \Sigma$, la transición $\delta(q, a)$ puede consistir en más de un estado o puede no estar definida. Más concretamente, un **autómata finito no-determinista** (AFN) está definido por

$$M = (\Sigma, Q, q_0, F, \Delta)$$

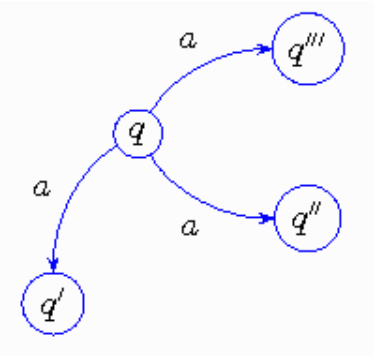
donde

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. $$\begin{array}{rcl} \Delta : Q \times \Sigma & \rightarrow & \wp(Q) \\ (q, a) & \mapsto & \Delta(q, a) = \{q_{i_1}, q_{i_2}, \ldots, q_{i_k}\} \end{array}$$
   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 palabra 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 aristas con la misma etiqueta:



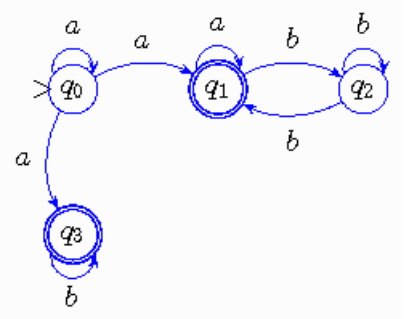
Un AFN M puede procesar una palabra de entrada $u \in \Sigma^*$ de varias maneras. Sobre el diagrama de estados del autómata, esto significa que pueden existir varias trayectorias etiquetadas con los símbolos de u.

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

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

Es decir, para que una palabra 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.

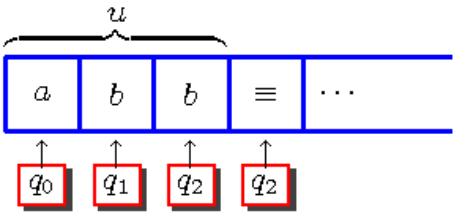
Ejemplo Sea M el siguiente AFN:



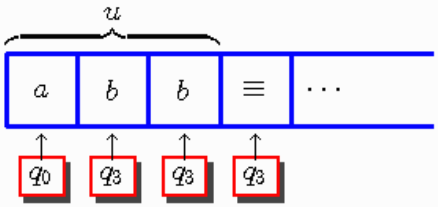
Δ	a	b
q_0	$\{q_0, q_1, q_3\}$	$\varnothing$
q_1	$\{q_1\}$	$\{q_2\}$
q_2	$\varnothing$	$\{q_1, q_2\}$
q_3	$\varnothing$	$\{q_3\}$

Para la palabra 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, $u \in L(M)$.

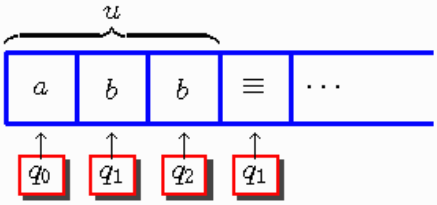
Cómputo de rechazo:



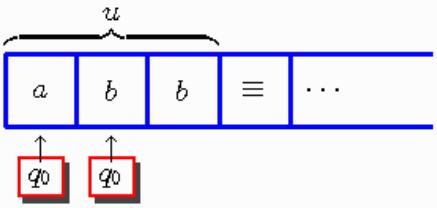
Cómputo de aceptación:



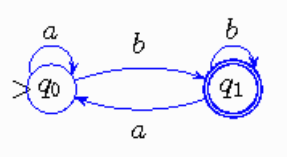
Otro cómputo de aceptación:



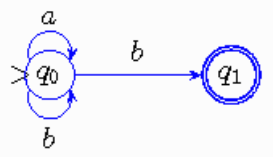
Cómputo abortado:



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

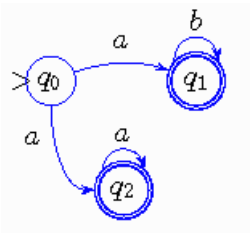


Un AFN que acepta el mismo lenguaje y que es, tal vez, más fácil de concebir, es el siguiente:

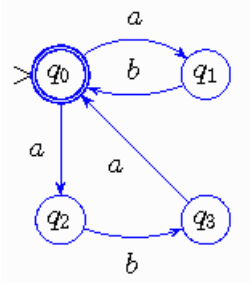


Este autómata se asemeja a la expresión regular $(a \cup b)^* b$.

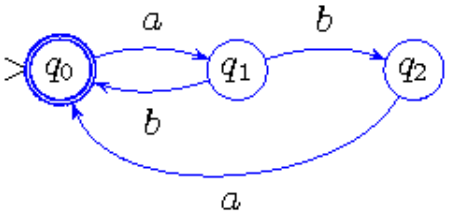
Ejemplo Considérese el lenguaje $L = ab^* \cup a^+$ sobre el alfabeto $\Sigma = \{a, b\}$. El siguiente AFN M satisface $L(M) = L$.



Ejemplo $\Sigma = \{a, b\}, L = (ab \cup aba)^*$. El siguiente AFN acepta a L.



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



Ejercicios Diseñar AFD's o AFN's que acepten los siguientes lenguajes:

1. $\Sigma = \{a, b, c\}$, $L =$ lenguaje de las palabras sobre Σ que no contienen la cadena bc. Véase una expresión regular para L en la sección 1.14.
2. $\Sigma = \{a, b\}, L = ab^+a^*$.
3. $\Sigma = \{a, b\}, L = a(a \cup ab)^*$.
4. $\Sigma = \{a, b, c\}, L = a^*b^*c^*$.

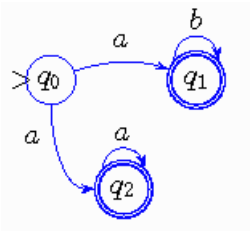
2.5. Equivalencia computacional entre los AFD y los AFN

En esta sección se mostrará que los modelos AFD y AFN son computacionalmente equivalentes. En primer lugar, es fácil ver que un AFD $M = (\Sigma, Q, q_0, F, \delta)$ puede ser considerado como un AFN $M' = (\Sigma, Q, q_0, F, \Delta)$ definiendo $\Delta(q, a) = \{\delta(q, a)\}$ para cada $q \in Q$ y cada $a \in \Sigma$. Para la afirmación recíproca tenemos el siguiente teorema:

Teorema 2.5.1. *Dado un AFN $M = (\Sigma, Q, q_0, F, \Delta)$ se puede construir un AFD M'* ***equivalente a*** *M, es decir, tal que $L(M) = L(M')$.*

Este teorema, cuya demostración se dará en detalle más adelante, establece que el no-determinismo se puede eliminar. Dicho de otra manera, los autómatas deterministas y los no deterministas aceptan los mismos lenguajes. La idea de la demostración consiste en considerar cada conjunto de estados $\{p_1, \ldots, p_j\}$ del autómata no-determinista como un *único* estado del nuevo autómata determinista. El siguiente ejemplo ilustra el procedimiento.

Ejemplo Consideremos el AFN M presentado en la sección 2.4, tal que $L(M) = ab^* \cup a^+$ sobre $\Sigma = \{a, b\}$:



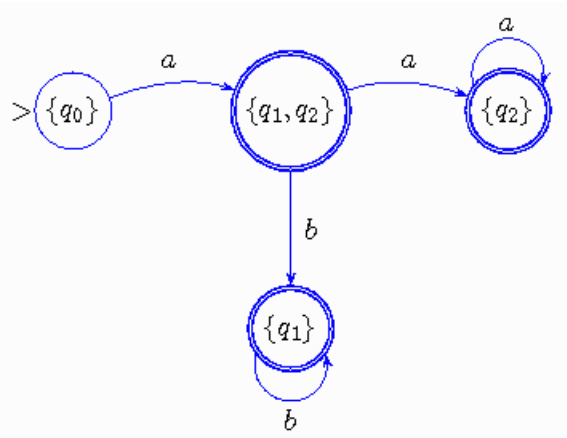
La función de transición Δ de M es:

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

El nuevo AFD M' construido a partir de M y equivalente a M tiene (por lo menos) un estado más: $\{q_1, q_2\}$ y su función de transición δ tiene el siguiente aspecto:

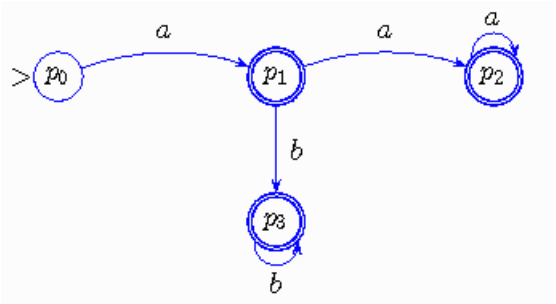
δ	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_1	$\varnothing$	$\{q_1\}$
q_2	$\{q_2\}$	$\varnothing$
$\{q_1, q_2\}$	$\{q_2\}$	$\{q_1\}$

El diagrama de estados de este autómata es:



Los estados de aceptación del nuevo autómata son los conjuntos de estados en los que aparece *por lo menos* un estado de aceptación del autómata original.

Para mayor simplicidad, podemos cambiar los nombres de los estados de este autómata:



Ejercicios Diseñar AFD's equivalentes a los ejemplos de AFN's construidos en la sección 2.4.

Para la demostración del teorema, conviene extender la definición de la función de transición, tanto de los autómatas determinista como de los no-deterministas.

Definición 2.5.1. Sea $M = (\Sigma, Q, q_0, F, \delta)$ un AFD. La función de transición $\delta : Q \times \Sigma \longrightarrow Q$ se extiende a una función $\widehat{\delta} : Q \times \Sigma^* \longrightarrow Q$ por medio de la siguiente definición recursiva:

$$\begin{aligned}
&\widehat{\delta}(q, \lambda) = q, \quad q \in Q, \\
&\widehat{\delta}(q, a) = \delta(q, a), \quad q \in Q,\ a \in \Sigma, \\
&\widehat{\delta}(q, wa) = \delta(\widehat{\delta}(q, w), a), \quad q \in Q,\ a \in \Sigma,\ w \in \Sigma^*.
\end{aligned}$$

Según esta definición, para una palabra de entrada $w \in \Sigma^*$, $\widehat{\delta}(q_0, w)$ es el estado en el que el autómata termina el procesamiento de w. Por lo tanto, podemos describir el lenguaje aceptado por M de la siguiente forma:

$$L(M) = \{w \in \Sigma^* : \widehat{\delta}(q_0, w) \text{ contiene un estado de aceptación}\}.$$

Notación. Sin peligro de ambigüedad, la función extendida $\widehat{\delta}(q, w)$ se notará simplemente $\delta(q, w)$.

Definición 2.5.2. ea $M = (\Sigma, Q, q_0, F, \Delta)$ un AFN. La función de transición $\Delta : Q \times \Sigma \longrightarrow \wp(Q)$ se extiende inicialmente a conjuntos de estados. Para $a \in \Sigma$ y $S \subseteq F$ se define

$$\Delta(S, a) := \bigcup_{q \in S} \Delta(q, a)$$

Podemos extender Δ a una función $\widehat{\Delta} : Q \times \Sigma^* \longrightarrow \wp(Q)$, de manera similar a como se hizo para AFD's. Recursivamente,

$$\begin{aligned}
&\widehat{\Delta}(q, \lambda) = \{q\}, \quad q \in Q, \\
&\widehat{\Delta}(q, a) = \Delta(q, a), \quad q \in Q,\ a \in \Sigma, \\
&\widehat{\Delta}(q, wa) = \Delta(\widehat{\Delta}(q, w), a) = \bigcup_{p \in \widehat{\Delta}(q,w)} \Delta(p, a), \quad q \in Q,\ a \in \Sigma,\ w \in \Sigma^*.
\end{aligned}$$

Según esta definición, para una palabra de entrada $w \in \Sigma^*$, $\widehat{\delta}(q_0, w)$ es el conjunto de los posibles estados en los que terminan los cómputos *completos* de w. Si el cómputo se aborta durante el procesamiento de w, se tendría $\widehat{\Delta}(q_0, w) = \varnothing$. Podemos describir el lenguaje aceptado por M de la siguiente forma:

$$L(M) = \{w \in \Sigma^* : \widehat{\Delta}(q_0, w) \text{ contiene un estado de aceptación}\}.$$

Notación. Sin peligro de ambigüedad, la función extendida $\widehat{\Delta}(q, w)$ se notará simplemente $\Delta(q, w)$.

A continuación se hará la demostración del teorema 2.5.1
Demostración: Dado el AFN $M = (\Sigma, Q, q_0, F, \Delta)$, construimos el AFD M' así:

$$M' = (\Sigma, \wp(Q), \{q_0\}, F', \delta)$$

donde

$$\begin{array}{rcl} \delta : \wp(Q) \times \Sigma & \to & \wp(Q) \\ (S, a) & \mapsto & \delta(S, a) := \Delta(S, a). \end{array}$$

$$F' = \{S \subseteq \wp(Q) : S \cap F \neq \varnothing\}.$$

Se demostrará que $L(M) = L(M')$ probando que, para toda palabra $w \in \Sigma^*$,

$$\delta(\{q_0\}, w) = \Delta(q_0, w).$$

La anterior igualdad se demostrará por inducción sobre w.
Para $w = \lambda$, claramente se tiene $\delta(\{q_0\}, \lambda) = \Delta(q_0, \lambda) = \{q_0\}$.
Para $w = a$, $a \in \Sigma$, se tiene

$$\delta(\{q_0\}, a) = \Delta(\{q_0\}, a) = \Delta(q_0, a).$$

Supóngase (hipótesis de inducción) que $\delta(\{q_0\}, w) = \Delta(q_0, w)$, y que $a \in \Sigma$.

$$\begin{array}{rcll} \delta(\{q_0\}, wa) & = & \delta(\delta(\{q_0\}, w), a) & \text{(definición de la extensión de } \delta) \\ & = & \delta(\Delta(\{q_0\}, w), a) & \text{(hipótesis de inducción)} \\ & = & \Delta(\Delta(\{q_0\}, w), a) & \text{(definición de } \delta) \\ & = & \Delta(\{q_0\}, wa) & \text{(definición de la extensión de } \Delta) \\ & = & \Delta(q_0, wa) & \text{(definición de la extensión de } \Delta) \end{array}$$

Esto demuestra el teorema.