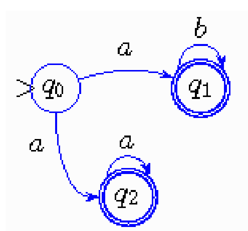
2.5. Equivalencia computacional entre los AFD y los AFN

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:

2.5.1 Teorema. *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 $\Sigma = \{a, b\}$:



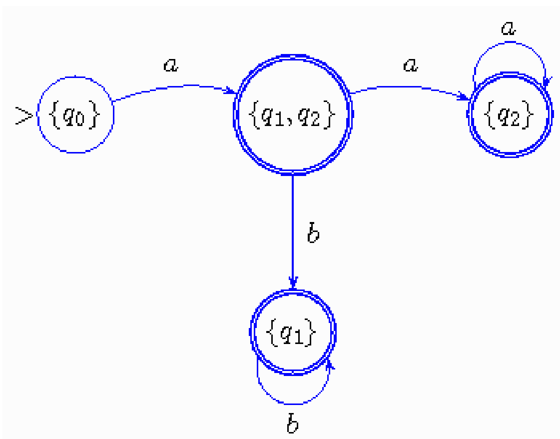
La función de transición Δ de M es:

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:

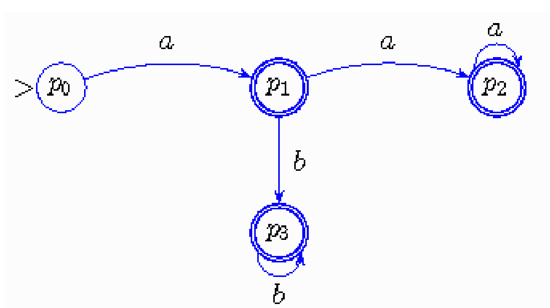
δ	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_1	$\varnothing$	$\{q_1\}$
q_2	$\{q_2\}$	$\varnothing$
$\{q_1, q_2\}$	$\{q_2\}$	$\{q_1\}$

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.

2.5.2 Definición. 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 $\delta(q, w)$.

2.5.3 Definición. 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
<u>*Demostración*</u>: 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.