5.1.5 Autómatas Finitos Deterministas: DFAs. Simulación de un DFA.

Definición: Sea $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ un FA, decimos que es **determinista** sii $|\delta(q,a)| \leq 1$, $\forall q \in Q, a \in \Sigma$.

Teorema: Sea $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ un NFA, entonces existe $\mathcal{A}' = (Q', \Sigma', \delta', q_0', F') \;/\; \underset{\mathcal{A}' \text{ DFA}}{T(\mathcal{A}') = T(\mathcal{A})}$.

demo.

Basta considerar
$$\begin{cases} Q' := \mathcal{P}(Q) \\ \Sigma' := \Sigma \\ F' := \{ S \subseteq Q \;/\; S \cap F \neq \emptyset \} \\ q_0' = \{q_0\} \\ \forall\, S \subseteq Q,\ \delta'(S,a) := S' \text{ donde:} \\ \qquad S' := \{ p \in Q \;/\; \exists q \in S,\ p \in \delta(q,a) \} \end{cases}$$

entonces para demostrar que $T(\mathcal{A}') = T(\mathcal{A})$, bastará probar que

$$\forall i,\ (S,u) \overset{i}{\underset{\mathcal{A}'}{\vdash}} (S',w) \Leftrightarrow S' = \{ p \in Q \;/\; \exists q \in S,\ (q,u) \overset{i}{\underset{\mathcal{A}}{\vdash}} (p,w) \}$$

Lo haremos por inducción en el valor de "i":

$i=1$

$$(S,uw) \overset{1}{\underset{\mathcal{A}'}{\vdash}} (S',w) :\Leftrightarrow \delta'(S,u) = S' :\Leftrightarrow S' = \{ p \in Q \;/\; \exists q \in S,\ (q,uw) \overset{1}{\underset{\mathcal{A}}{\vdash}} (p,w) \}$$

$i=n-1$ Supuesto cierto.

$i = n$

$$(S, uw) \overset{n}{\underset{A'}{\vdash}} (S', w) :\Leftrightarrow (S, uw) \overset{n-1}{\underset{A'}{\vdash}} (S'', v) \underset{A'}{\vdash} (S', w) \Leftrightarrow$$

$$\Leftrightarrow \begin{cases} S'' = \{ t \in Q / \exists q \in S, \ (q, uw) \overset{n-1}{\underset{A}{\vdash}} (t, v) \} \ \text{(por inducción)} \\ S' = \{ p \in Q / \exists t \in S'', \ (t, v) \underset{A}{\vdash} (p, w). \ \text{(por definición)} \end{cases}$$

$$:\Leftrightarrow S' = \{ p \in Q / \exists q \in S, \ (q, uw) \overset{n}{\underset{A}{\vdash}} (p, w) \}$$

Por tanto, tendremos que en particular

$$x \in T(A') \Leftrightarrow (\{q_0\}, x) \overset{i}{\underset{A'}{\vdash}} (q'_j, \varepsilon) \Leftrightarrow \exists p \in F / (q_0, x) \overset{i}{\underset{A}{\vdash}} (p, \varepsilon) \Leftrightarrow : x \in T(A) \quad (q'_j \in F')$$

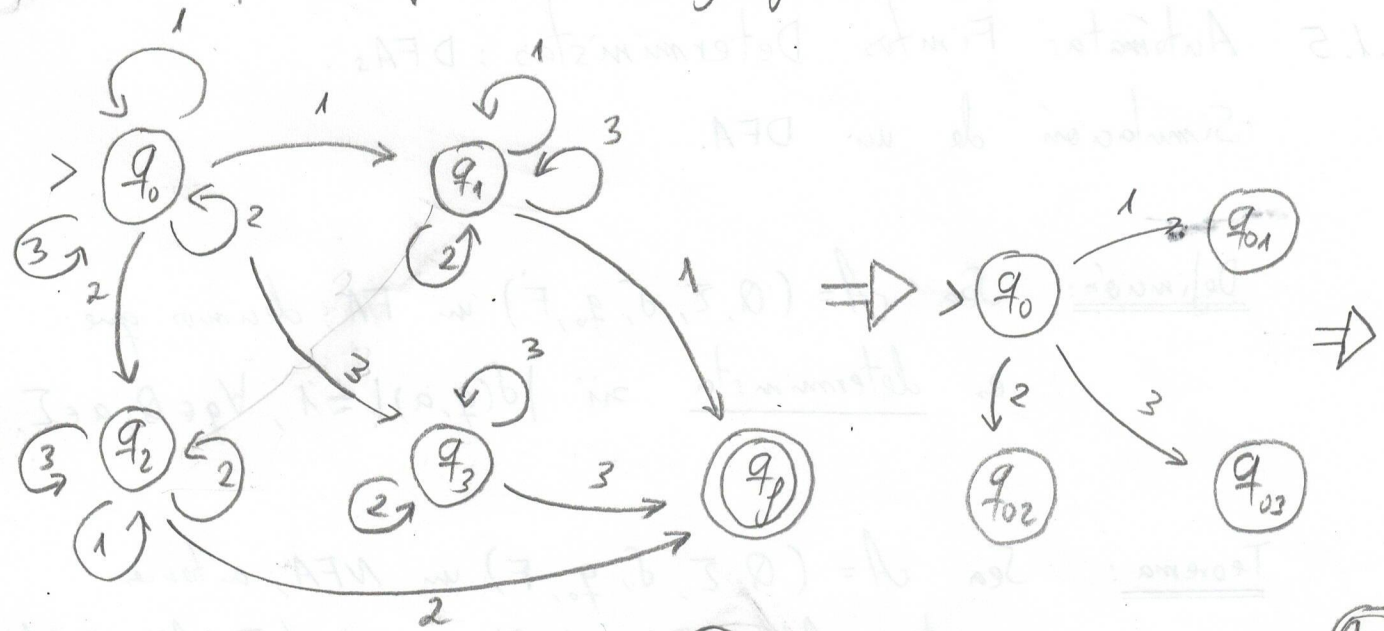
en conclusión $T(A') = T(A)$. demostrado

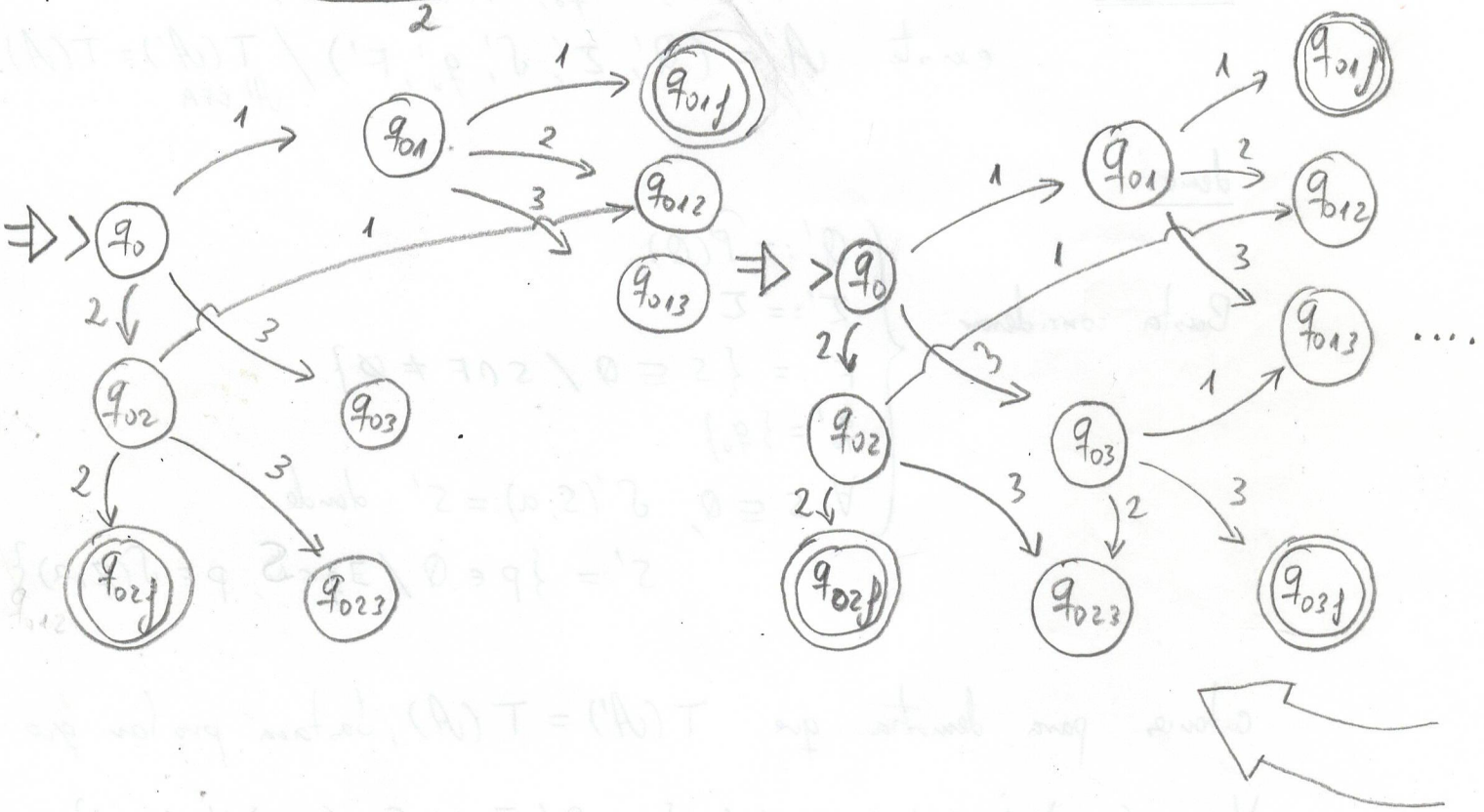
Ejemplo: Sea $A = (\{q_0, q_1, q_2, q_3, q_f\}, \{1, 2, 3\}, \delta, q_0, \{q_f\})$ donde δ viene dada por:

δ	1	2	3
q_0	$\{q_0, q_1\}$	$\{q_0, q_2\}$	$\{q_0, q_3\}$
q_1	$\{q_1, q_f\}$	$\{q_1\}$	$\{q_1\}$
q_2	$\{q_2\}$	$\{q_2, q_f\}$	$\{q_2\}$
q_3	$\{q_3\}$	$\{q_3\}$	$\{q_3, q_f\}$
q_f	$\emptyset$	$\emptyset$	$\emptyset$

construiremos $A' = (Q', \{1, 2, 3\}, \delta', \{q_0\}, F)$ DFA / $T(A') = T(A)$.

... los primeros pasos ilustrados gráficamente:





...

En principio basta tomar A' en la forma indicada en el Th. anterior, con lo que $Q' := \mathcal{P}(Q)$ y por tanto $|Q'| = 2^5 = 32$. Sin embargo, veremos que el número de estados puede reducirse considerablemente eliminando aquellos que no son accesibles.

Definición: Sea $A = (Q, \Sigma, \delta, q_0, F)$ un FA, decimos que $p \in Q$ es un estado accesible sii $\exists w /\ (q_0, w) \vdash^* (p, e)$.

Es evidente que en un FA, sólo es necesario considerar los estados que son accesibles.

En nuestro caso, y aplicando la definición de accesible, obtenemos que las tablas de transición de A' pueden reducirse a:

	1	2	3
$A = \{q_0\}$	B	C	D
$B = \{q_0, q_1\}$	E	F	G
$C = \{q_0, q_2\}$	F	H	I
$D = \{q_0, q_3\}$	G	I	J
$E = \{q_0, q_1, q_p\}$	E	F	G
$F = \{q_0, q_1, q_2\}$	K	K	L
$G = \{q_0, q_1, q_3\}$	M	L	M
$H = \{q_0, q_2, q_p\}$	F	H	I
$I = \{q_0, q_2, q_3\}$	L	N	N
$J = \{q_0, q_3, q_p\}$	G	I	J
$K = \{q_0, q_1, q_2, q_p\}$	K	K	L
$L = \{q_0, q_1, q_2, q_3\}$	P	P	P
$M = \{q_0, q_1, q_3, q_p\}$	M	L	M
$N = \{q_0, q_2, q_3, q_p\}$	L	N	N
$P = \{q_0, q_1, q_2, q_3, q_p\}$	P	P	P

Por tanto, el estado inicial de A' es A.
los estados finales de A' son E, H, J, K, M, N, P.

Teorema: Sea $A = (Q, \Sigma, \delta, q_0, F)$ un DFA y sea $x \in \Sigma^*$, entonces el test $x \in T(A)$ puede realizarse en un tiempo $O(|x|)$.

demo.

Consideremos el algoritmo siguiente, simulando un DFA.

```
s := q0;
c := leer_siguiente_caracter;
WHILE c ≠ EOF DO BEGIN
        s := δ(s, c);
        c := leer_siguiente_caracter;
END;
IF s ∈ F THEN
    RETURN "yes"
ELSE RETURN "no"
```

Es evidente que ello permite establecer el test $x \in T(A)$, además la complejidad del bucle WHILE es trivialmente $O(|x|)$.

demostrado