UNIDAD II: LENGUAJES REGULARES

Capítulo 4. Gramáticas regulares.

Capítulo 5. Autómatas finitos deterministas

5.1. Concepto de AFD

Definición. Lenguaje reconocido por un AFD.

5.2. Equivalencia y Minimización de AFD

Accesibilidad. Equivalencia de estados. AFD mínimo. Equivalencia de autómatas.

Capítulo 6. Autómatas finitos no deterministas

Capítulo 7. Expresiones regulares

Capítulo 8. Propiedades de lenguajes regulares

Capítulo 9. Otros tipos de autómatas

5.1 Concepto de AFD

- Un AFD es una máquina de estados que tiene acceso a una secuencia de símbolos de entrada (mediante una cabeza lectora).
- Un AFD se encuentra en cada momento en un estado determinado y puede transitar a otro estado. Para ello se realizan los siguientes pasos:
  - Se lee la cinta y se avanza la cabeza lectora.
  - En función del símbolo leído y del estado actual el autómata transita a otro estado
- Un AFD para el procesamiento cuando no le quedan más símbolos en la entrada
- La parada puede ocurrir en un estado marcado como "final" o uno que no este marcado como final

  $\rightarrow$ "salida binaria"

5.1.1 Definición

Un autómata finito determinista (AFD) es una quíntupla $A=(Q,\Sigma,f,q_0,F)$, donde:

- Q es un conjunto de estados
- Σ es el alfabeto de entrada
- $f:Q\times\Sigma\rightarrow Q$ es la función (total) de transición
- $q_0 \in Q$ es el estado inicial
- $F\subseteq Q$ es el conjunto de estados finales

Ejemplo: (sencillo: Autómata que acepta las cadenas $(01)^n1$):

A=($\{q_0,q_1,q_2,q_3\}$,$\{0,1\}$,f,q_0,$\{q_2\}$), donde la función de transición f está definida como sigue a continuación:

$f(q_0,0)=q_1$ $f(q_0,1)=q_2$
$f(q_1,0)=q_3$ $f(q_1,1)=q_0$
$f(q_2,0)=q_3$ $f(q_2,1)=q_3$
$f(q_3,0)=q_3$ $f(q_3,1)=q_3$

Alternativamente, también podemos representar el autómata (no sólo la función de transición) mediante una *tabla de transiciones*:

	A	0	1
→	q_0	q_1	q_2
	q_1	q_3	q_0
*	q_2	q_3	q_3
	q_3	q_3	q_3

El hecho de que f sea una función *total* significa que está definida para todo par $(q,a)\in Q\times\Sigma$.

El *determinismo* del autómata proviene del hecho de que la imagen de la función de transición sea Q. Es decir, en cada momento (para cada símbolo de entrada y cada estado) sólo tiene definido una posible transición.

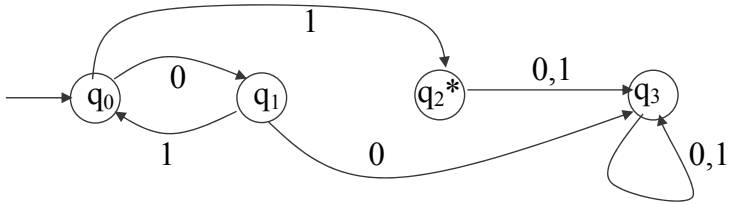
El nombre de *finito* proviene del hecho de que el autómata sólo tiene un conjunto finito de estado distintos para recordar lo procesado (no tiene ningún sistema de almacenamiento de información adicional).

Definición:
Sea un autómata A=(Q,Σ,f,q_0,F). El *diagrama de transición de estados* de dicho autómata es el grafo que cumple las siguientes propiedades:

- El conjunto de nodos del grafo es Q
- El nodo correspondiente al estado inicial (y sólo él) está marcado con una flecha
- Los |F| nodos correspondientes a los estados finales (y sólo ellos) están marcados mediante "*" o mediante un doble círculo
- Existe un arco desde el nodo q_i al q_j etiquetado mediante el símbolo a si y sólo si $f(q_i,a)=q_j$.

Ejemplos:

1. Grafo de transición para el mismo autómata del ejemplo anterior:



2. Grafos sencillos que **no** corresponden a autómatas finitos deterministas:
   - carecen de transiciones para determinados estados y símbolos del alfabeto
   - algunas transiciones no están etiquetadas
   - no tiene estado inicial (un AFD **si** puede carecer de estados finales)

5.1.2 Lenguaje reconocido por un AFD

Definición:
La *función de transición extendida* $f^*: Q\times\Sigma^*\to Q$, se define recursivamente de la siguiente manera:

- $f^*(q,\lambda)=q$, para todo $q \in Q$
- $f^*(q,ax)=f^*(f(q,a),x)$, para todo $q \in Q$, $a \in \Sigma$ y $x \in \Sigma^*$

Nota: la segunda parte también se puede definir como
$f^*(q,xa)=f(f^*(q,x),a)$

Ejemplo:
Teniendo en cuenta el autómata del ejemplo anterior:
¿Qué valores tienen $f^*(q_0,0101)$, $f^*(q_0,(01)^31)$?
¿Qué valor tienen $f^*(q_0,(01)^n1)$ para todo $n\geq 0$?

Si $x \neq (01)^n1$, entonces $f^*(q_0,x) \neq q_2$

Definición (palabra aceptada):
Sea un autómata $A=(Q,\Sigma,f,q_0,F)$. Se dice que el autómata A *acepta* una palabra $x\in\Sigma^*$ si y sólo si: $f^*(q_0,x) \in F$.

Definición (lenguaje reconocido):
Sea un autómata $A=(Q,\Sigma,f,q_0,F)$. El lenguaje *reconocido* (o *aceptado*) por dicho autómata está formado por el conjunto de palabras que acepta:

$$L(A)=\{x \mid x\in\Sigma^* \text{ y } f^*(q_0,x) \in F\}.$$

Definición (AFDs equivalentes):
Dos autómatas A_1 y A_2 se dicen equivalentes ($A_1\equiv A_2$), si reconocen el mismo lenguaje:

$$L(A_1)=L(A_2).$$

El complemento del lenguaje aceptado por un autómata está formado por el conjunto de palabras que hacen terminar al autómata en un estado que no es final:

$$\overline{L(A)}=\{x \mid x\in\Sigma^* \text{ y } f^*(q_0,x) \notin F\}.$$

El lenguaje reconocido por un autómata incluye λ si y sólo si el estado inicial es un estado final:

$$q_0\in F \Leftrightarrow \lambda \in L(A)$$

Ejemplos:

1. Si Q=F, entonces $L(A)=\Sigma^*=W(\Sigma)$
2. El autómata más pequeño tal que $L(A)=\{a,b\}^*$:

	A	a	b
$\to^*$	q_0	q_0	q_0

3. Si $F=\varnothing$, entonces $L(A)=\varnothing$.
4. El autómata más pequeño tal que $L(A)=\varnothing$ (con $\Sigma=\{a,b\}$):

	A	a	b
$\to$	q_0	q_0	q_0

5. Para el autómata del primer ejemplo: $L(A)=\{(01)^n1 \mid n\geq 0\}$
6. Encontrar un AFD que acepte el lenguaje siguiente:
   $L=\{x \mid x=aby \text{ e } y\in\{a,b\}^*\}$
   Los cuatro estados del autómata se pueden denominar: "λ", "a", "ab..." y "...".
7. Encontrar un AFD que acepte el lenguaje siguiente:
   $L=\{x \mid x\in\{0,1\}^* \text{ y } x \text{ no contiene el substring } 001\}$
   El autómata tiene 4 estados, que podemos denominar "λ", "0", "00" y "001".
8. Encontrar un AFD que acepte el lenguaje siguiente:
   $L=\{x \mid x\in\{a,b\}^* \text{ y el número de a's es múltiplo de } 3\}$

5.2 Equivalencia y minimización de AFD

· Se puede construir diferentes autómatas que aceptan el mismo lenguaje.
  ⇒ Resulta interesante encontrar el autómata más sencillo que reconozca un determinado lenguaje.
  ⇒ Minimización de AFD

La minimización de un autómata pasa, normalmente, por la minimización del conjunto de estados. Hay dos formas de reducir los estados:
· eliminando estados inaccesibles,
· combinando estados equivalentes.

5.2.1 Accesibilidad

Definición (estado accesible):
Sea un AFD $A=(Q,\Sigma,f,q_0,F)$. Se dice que un estado $q_i \in Q$ es *accesible* desde otro estado $q_j \in Q$ si: $\exists\, x \in \Sigma^*$ tal que $f^*(q_i,x)=q_j$.

Todo estado es accesible desde sí mismo ya que $f^*(q,\lambda)=q$.

Si ninguno de los estados finales es accesible desde el inicial, entonces $L(A)=\varnothing$.

Definición (autómata conexo):
Sea un AFD $A=(Q,\Sigma,f,q_0,F)$. Se dice que el autómata A es *conexo* si: $\forall\, q \in Q$: q es accesible desde q_0.

Ejemplos:
1. (autómata inconexo, con vértices aislados en el diagrama de transiciones)
   $A_1=(\{p,q,r\},\{a\},f,p,\{q,r\})$, con la siguiente tabla de transiciones:

	A_1	a
→	p	r
*	q	q
*	r	r

2. (autómata inconexo, sin vértices aislados)
   $A_2=(\{p,q,r\},\{a\},f,p,\{r\})$, con la siguiente tabla de transiciones:

	A_2	a
→	p	r
	q	p
*	r	r

Si eliminamos de un autómata todos aquellos estados que no sean accesibles desde el estado inicial (es decir, si hacemos conexo un autómata que no lo era), el lenguaje aceptado por el nuevo autómata no cambiará.

Ejemplos (anteriores):
1. $A_1'=(\{p,r\},\{a\},f,p,\{r\})$

	A_1'	a
→	p	r
*	r	r

2. $A_2'=(\{p,r\},\{a\},f,p,\{r\})$

	A_2'	a
→	p	r
*	r	r

Se puede ver que los dos autómatas en realidad son equivalentes:
$L(A_1)=L(A_1') \wedge L(A_2)=L(A_2') \wedge A_1'=A_2' \Rightarrow L(A_1)=L(A_2)$

Algoritmo (encontrar autómata conexo):
Trivial: considera el siguiente ejemplo

	A	a	b
→	p	r	q
*	r	r	p
	q	r	q
	s	r	t
	t	s	P

El resultado del algoritmo es un autómata conexo que es igual que A, con la única diferencia de que en el se han eliminado los estados inaccesibles desde el estado q_0.

Lema 1:
El autómata A' que se obtiene como resultado del algoritmo anterior es equivalente al autómata A y tiene un número de estados menor o igual a A.

Demostración:
- Para cualquier palabra $x \in L(A)$ y cualquier prefijo z de x (x=zy) se cumple que $f^*(q_0,z)$ es un estado accesible desde q_0.
- Luego la aceptación de x sólo pasa por estados accesibles desde q_0.
- Por tanto, la eliminación de los estados inaccesibles no influye en la aceptación de palabras, por lo que L(A')=L(A).
- Es obvio que la eliminación de estados inaccesibles no incrementa el número de estados del autómata.

5.2.2 Equivalencia de estados

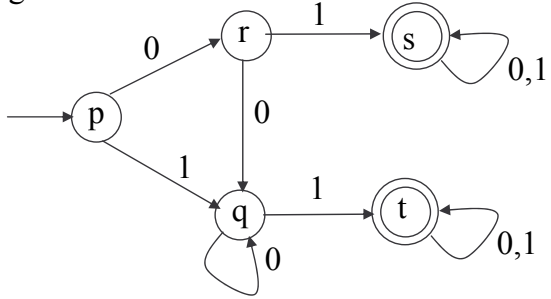
Definición (estados equivalentes):
- Dos estados p y q se dicen *equivalentes* (o *indistinguibles*), pEq, si: $\forall x \in \Sigma^*: f^*(p,x) \in F \Leftrightarrow f^*(q,x) \in F$.
- Dos estados p y q se dicen *equivalentes* (o *indistinguibles*) *respecto a palabras de longitud n o menor*, pE_nq, si:
$\forall x \in \Sigma^*$ con $|x| \leq n$: $f^*(p,x) \in F \Leftrightarrow f^*(q,x) \in F$.

Definición (estados distinguibles):
Dos estados p y q se dicen distinguibles si no son equivalentes.

Ejemplos:
1. Ningún estado equivalente, todos distinguibles:

	A	a
→	q_0	q_1
	q_1	q_2
	q_2	q_3
*	q_3	q_0

2. Estados equivalentes:
A=({p,r,q,s,t},{0,1},f,p,{s,t}) con f definido por el siguiente grafo de transición:



Lema 2:
Las relaciones de indistinguibilidad de estados E (y E_n) son relaciones de equivalencia.

Definición: [Matemática Discreta, Cap. 3, p.21]
Sea A un conjunto y R una relación de equivalencia en A. Para cada elemento $a \in A$ se define la *clase de equivalencia* de a, y se denota por C(a) o por $\bar{a}$, como el conjunto de todos aquellos elementos de A que se relacionan con a:

$$C(a)=\{b \mid b \in A \text{ y } aRb\}.$$

Definición: [Matemática Discreta, Cap. 3, p.22]
Sean A un conjunto y R una relación de equivalencia en A. Se define el *conjunto cociente de A por la relación de equivalencia R*, como el conjunto de las clases de equivalencia:

$$A/R = \{C(a) \mid a \in A\}.$$

El conjunto cociente de Q por la relación E (E_n), representado por Q/E (Q/E_n), es una partición de Q en clases de estados equivalentes (equivalentes para palabras de longitud n).

Lema 3:
Dado un autómata $A=(Q,\Sigma,f,q_0,F)$ se cumple $Q/E_0=\{F,Q-F\}$.

Demostración:
Sea $|x|\leq 0$ ($x=\lambda$). Por definición $f^*(q,\lambda)=q$ para cualquier estado q.
Obviamente, para todo $q \in F$ y sólo para ellos: $f^*(q,\lambda) \in F$.
Por otro lado, para todo $p \notin F$, y sólo para ellos: $f^*(p,\lambda) \in Q-F$.

Lema 4:
Sea un autómata $A=(Q,\Sigma,f,q_0,F)$, y dos estados p y $q \in Q$.

$$\forall a \in \Sigma: f(p,a)E_n f(q,a) \Leftrightarrow pE_{n+1}q.$$

Demostración:
$\forall a \in \Sigma: f(p,a)E_n f(q,a)$
$= \quad \forall a \in \Sigma$ y $\forall x \in \Sigma^*$ con $|x|\leq n$: $f^*(f(p,a),x) \in F \Leftrightarrow f^*(f(q,a),x) \in F$
$= \quad \forall a \in \Sigma$ y $\forall x \in \Sigma^*$ con $|x|\leq n$: $f^*(p,ax) \in F \Leftrightarrow f^*(q,ax) \in F$
$= \quad \forall y \in \Sigma^*$ con $|y|\leq n+1$: $f^*(p,y) \in F \Leftrightarrow f^*(q,y) \in F$
$= \quad pE_{n+1}q$

Lema 5:
Si $Q/E_n = Q/E_{n+1}$, entonces $Q/E_n = Q/E_{n+i}$ para i=0,1,2... (=Q/E).

Demostración: (por inducción)
Base: evidente para i=0 y i=1
Paso inductivo:
Suposición: $Q/E_n = Q/E_{n+1} = ... = Q/E_{n+j}$
Queda demostrar que: $Q/E_n = Q/E_{n+1} = ... = Q/E_{n+j+1}$

Sean p y q cualesquiera dos estados con $pE_{n+j}q$.

- Del lema 4 se sigue: $\forall a \in \Sigma: f(p,a)E_{n+j-1}f(q,a)$, es decir para todo a, f(p,a) y f(q,a) pertenecen a la misma clase en Q/E_{n+j-1}
- Dado que $Q/E_{n+j} = Q/E_{n+j-1}$, también pertenecen a la misma clase en Q/E_{n+j}, es decir, $\forall a \in \Sigma: f(p,a)E_{n+j}f(q,a)$.
- Luego, aplicando el lema 4 otra vez: $pE_{n+j+1}q$
- Por tanto se verifica $Q/E_{n+j} = Q/E_{n+j+1}$.

Los lemas anteriores permiten construir un algoritmo para encontrar el conjunto cociente a partir de Q/E_0.

Algoritmo:
Dado un autómata $A=(Q,\Sigma,f,q_0,F)$, obtener el conjunto cociente Q/E:

1. $Q/E_0=\{F,Q-F\}$.
2. Generar Q/E_{i+1} a partir de Q/E_i:
   p y q pertenecen a la misma clase en Q/E_{i+1} sii:
   - p y q pertenecen a la misma clase en Q/E_i, y
   - para todo $a\in\Sigma$, f(p,a) y f(q,a) pertenecen a la misma clase en Q/E_i
3. Si $Q/E_{i+1}=Q/E_i$, entonces $Q/E= Q/E_i$. En otro caso, volver al paso 2.

Ejemplo:
(ejemplo de antes) $A=(\{p,r,q,s,t\},\{0,1\},f,p,\{s,t\})$ con la tabla de transición:

	A	0	1
→	p	r	q
	r	q	s
	q	q	t
*	s	s	s
*	t	t	t

- $Q/E_0=\{\{p,r,q\},\{s,t\}\}$
- estados que irán juntos en Q/E_1:
  pEr-no(x=1); pEq-no(x=1); rEq-¿?; sEt-¿?
- $Q/E_1=\{\{p\},\{r,q\},\{s,t\}\}$
- estados que irán juntos en Q/E_2:
  rEq-¿?; sEt-¿?
- $Q/E_2=\{\{p\},\{r,q\},\{s,t\}\}= Q/E_1=Q/E$

NOTA:

Como ayuda, en cada paso de Q/E_i a Q/E_{i+1}, se puede escribir otra tabla en la que se sustituyen los estados por las clases de Q/E_i en las columnas 2 y seguidas. Entonces, dos estados que pertenecen a la misma clase en Q/E_i, pertenecen también a la misma clase en Q/E_{i+1}, sii para todo símbolo $a\in\Sigma$ se las transiciones llevan a la misma clase de Q/E_i.

Para el ejemplo anterior:

- $Q/E_0=\{\{p,r,q\},\{s,t\}\}$
- Paso de Q/E_0 a Q/E_1: se define {p,q,r}=C y {s,t}=D:

	A	0	1
→	p	C	C
	r	C	D
	q	C	D
*	s	D	D
*	t	D	D

→ $Q/E_1 = \{\{p\},\{r,q\},\{s,t\}\}$

- Paso de Q/E_1 a Q/E_2: se define {p}=B; {q,r}=C y {s,t}=D:

	A	0	1
→	p	C	C
	r	C	D
	q	C	D
*	s	D	D
*	t	D	D

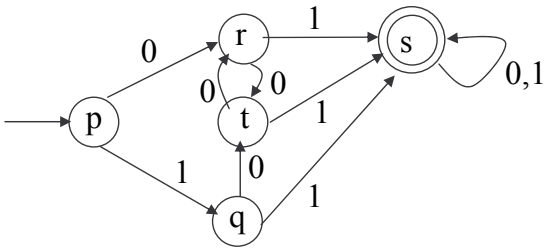
→ $Q/E_2 = \{\{p\},\{r,q\},\{s,t\}\}$

Ejemplos:

1. Calcula el conjunto cociente para el siguiente AFD.

	AFD	0	1
→	A	B	F
	B	G	C
*	C	A	C
	D	C	G
	E	H	F
	F	C	G
	G	G	E
	H	G	C

2. Calcula el conjunto cociente para el AFD definido por el siguiente grafo:



Para simplificar un autómata se pueden "unificar" los estados equivalentes. ¿Cómo sería el AFD con estados "unificados" correspondiente al ejemplo 2?

Lema 6:

Sea $A=(Q,\Sigma,f,q_0,F)$ un AFD. El autómata $A'=(Q/E,\Sigma,f',q_0',F')$, donde:

1. q_0' es el elemento de Q/E, tal que $q_0 \in q_0'$
2. $F'=\{s \mid s \in Q/E \text{ y } \exists\, p \in s: p \in F\}$
3. $f'(s_i,a)=s_j$ sii $\exists\, p \in s_i$ y $q \in s_j$ tal que $f(p,a)=q$

es equivalente a A.

Idea de demostración:

- El autómata A' tiene estados que corresponden a las clases de equivalencias del autómata A.
- Se demuestra que cualquier transición en el autómata A de un estado p a otro q tiene una transición equivalente en el autómata A' de la clase de equivalencias que contiene p a la clase de equivalencias que contiene q.
- Por tanto, dada cualquier palabra $x \in \Sigma^*$, si A para con x en un estado p, A' para con x en la clase de equivalencias que contiene p. Estos dos estado o ambos son finales o ambos no son finales.

5.2.3 AFD mínimo

Es obvio que tanto la eliminación de estados inaccesibles como la unificación de estados equivalentes pueden reducir el número de estados de un autómata (como mucho lo mantienen).

Algoritmo:
Dado A=(Q,Σ,f,q_0,F), obtener el AFD mínimo equivalente:

1. Eliminar todos los estados que no sean accesibles desde q_0
2. Construir el conjunto cociente Q/E
3. El autómata mínimo es A'=(Q/E,Σ,f',q_0',F'), donde:
   a. q_0' es el elemento de Q/E, tal que $q_0 \in q_0$'
   b. F'={s | s∈ Q/E y ∃ p∈ s tal que p∈ F}
   c. f'(s_i,a)=s_j sii ∃ p∈ s_i y q∈ s_j tal que f(p,a)=q

Ejemplo:
1. Obtener el autómata mínimo equivalente al siguiente AFD:

	AFD	0	1	2
→	p	r	t	q
	q	q	v	p
	r	p	u	r
*	s	q	t	u
*	t	t	v	u
*	u	t	t	v
*	v	u	u	t

Lema 7:
El autómata A' que es obtiene con el algoritmo anterior es equivalente al autómata A y no tiene ni estados inaccesibles ni estados equivalentes.

Este lema sigue de forma directa de los lemas 1 y 6 y del propio algoritmo.

Teorema 1:
Cualquier autómata A que no tiene estados inaccesibles desde el estado inicial ni estados equivalentes es mínimo (no existe ningún autómata con un número de estados menor que reconoce el lenguaje L(A)).

Idea de demostración:
Sea el autómata A=(Q,Σ,f,q_0,F) con *n* estados.
Supongamos que existiese un autómata A'=(Q',Σ,f',q_0',F') equivalente con menos estados (|Q'|<n).

- Tiene que haber palabras x e y que llevan a diferentes estados (p y q) en A y a un mismo estado en A'.
- En A, p y q no son equivalentes, por tanto existe una palabra w tal que f*(w,p) es final y f*(w,q) no es final (o vice versa).
- Por tanto, A acepta la palabra x.w y rechaza y.w.
- Por otro lado considerando A', las palabras x e y llevan al mismo estado, por tanto también las palabras x.w e y.w llevan a un mismo estado. Es decir, A' o rechaza ambas o acepta ambas.
- Eso contradice la hipótesis de que A' y A son equivalentes, por lo que A' no puede existir.

5.2.4 Equivalencia de AFDs

Definición (autómata suma):

Sean dos autómata $A_1=(Q_1,\Sigma,f_1,q_{0\_1},F_1)$ y $A_2=(Q_2,\Sigma,f_2,q_{0\_2},F_2)$ tales que $Q_1 \cap Q_2=\varnothing$. Se llama *suma* de A_1 y A_2 al autómata $A=A_1+A_2=(Q_1\cup Q_2,\Sigma,f,q_0,F_1\cup F_2)$, donde:

- q_0 es uno cualquiera de los estados q_{0_1} y q_{0_2} y
- $f(q,a)=\begin{cases} f_1(q,a), \text{si } q\in Q_1 \\ f_2(q,a), \text{si } q\in Q_2 \end{cases}$

Teorema 2:
Sean A_1 y A_2 dos autómatas. $A_1\equiv A_2$, si sus respectivos estados iniciales son equivalentes en el autómata $A=A_1+A_2$.

Idea de demostración

- Por construcción, el autómata suma no es conexo. En él existen dos partes claramente separados y entre los que no existen conexiones (los dos autómatas A_1 y A_2).
- Si los dos estados iniciales de A_1 y A_2, q_{0_1} y q_{0_2}, son equivalentes en A, se verifica que:
  $\forall\ x\in\Sigma^*$: desde q_{0_1} con x se llega a un estado final sii desde q_{0_2} con x se llega a un estado final
- Dado que ambos autómatas, A_1 y A_2, "existen por separado" en A, eso implica que $\forall\ x\in\Sigma^*$ ambos aceptan x o bien ambos rechazan x. Por tanto son equivalentes.

Nota: Este teorema nos facilita un algoritmo para comprobar la equivalencia de autómatas.

Ejemplo:
1. Decide si los dos autómatas siguientes son equivalentes.

	A_1	a	b
$\rightarrow$	p	q	s
*	q	r	q
*	r	q	s
	s	s	s

	A_2	a	b
$\rightarrow$*	z	y	x
*	y	z	y
	x	x	x

Teorema 3:
Sean A_1 y A_2 dos autómatas. $A_1\equiv A_2$, sii sus autómatas mínimos son iguales salvo renombramiento de estados (son isomorfos).

Idea de demostración:
Sean A_1' y A_2' los dos autómatas mínimos, respectivamente. Por los resultados de antes se sabe que $L(A_1')=L(A_1)$ y $L(A_2')=L(A_2)$.
SOLO SI: Si A_1' y A_2' son iguales (salvo renombramiento de estados) se cumple $L(A_1')=L(A_2')$, y entonces $L(A_1)=L(A_2)$.
SI: Si $A_1\equiv A_2$, entonces A_1' y A_2' son iguales salvo renombramiento de estados.
Vale demostrar que para todo estado en A_1' existe uno y sólo un estado equivalente en A_2' (y viceversa).

Ejemplo:
1. Usa los mismos autómatas del ejemplo anterior con el único cambio de que tanto r en A_1 como z en A_2 no sean estados finales. Obtén los autómatas mínimo para A_1 y A_2 y compáralos.