Tema 3:
Gramáticas regulares

Teoría de autómatas y lenguajes formales I

Bibliografía

- Hopcroft, J. E., Motwani, R., y Ullman, J. D. “Introducción a la Teoría de Autómatas, Lenguajes y Computación”. Addison Wesley. 2002.
  - capítulos 3 y 4
- Sudkamp, Thomas A. “Languages and machines : an introduction to the theory of computer science”. Addison-Wesley Publishing Company, 1998.
  - capítulo 7

Expresiones regulares

- Descripción algebraica de los lenguajes regulares
- Forma declarativa de expresar las cadenas que queremos aceptar
- Se usan como lenguaje de entrada en muchos sistemas de proceso de cadenas
  - comando *grep* de UNIX
  - generadores de analizadores léxicos, como *Lex* o *Flex*
    - acepta expresiones regulares (formas de las unidades sintácticas) y produce un AFD que reconoce la unidad sintáctica que aparece a continuación en la entrada

Operadores de las ER

- Las expresiones regulares denotan lenguajes
  - $01^* + 10^*$
- Unión de dos lenguajes *L* y *M*, $L \cup M$: conjunto de cadenas que pertenecen a *L*, a *M* o a ambos
- Concatenación de dos lenguajes *L* y *M*, *L*.*M* (o *LM*): conjunto de cadenas formadas por una cadena de *L* y otra de *M*
- Clausura, estrella o clausura de Kleene de un lenguaje *L*, L^*: conjunto de cadenas formado por la concatenación de cualquier número de cadenas de *L* (se admiten repeticiones). Formalmente:

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

- Sólo existen dos lenguajes con clausura no infinita:
  - L = {∅}. L^* = {**ε**}, ya que $\varnothing^0$ = {**ε**}, y $\varnothing^i$ = ∅ para i > 0
  - L = {**ε**}. L^* = {**ε**}

Precedencia de los operadores

- El operador * tiene la mayor precedencia
- El operador de concatenación (.) es el segundo en orden de precedencia
- Finalmente, se aplican los operadores de unión con sus operandos
- Utilizando los paréntesis se modifican las reglas de precedencia

Álgebra de ER

- Propiedad conmutativa de la unión: $L + M = M + L$
- Propiedad asociativa de la unión: $(L + M) + N = L + (M + N)$
- Propiedad asociativa de la concatenación: $(LM)N = L(MN)$
  - la concatenación no es conmutativa: $LM \neq ML$
- $\emptyset$ es el elemento identidad de la unión: $\emptyset + L = L + \emptyset = L$
- ε es el elemento identidad de la concatenación: $\varepsilon L = L \varepsilon = L$
- $\emptyset$ es el elemento nulo de la concatenación: $\emptyset L = L\emptyset = \emptyset$
- Propiedad distributiva por la izquierda de la concatenación respecto de la unión: $L(M + N) = LM + LN$
- Propiedad distributiva por la derecha de la concatenación respecto de la unión: $(M + N)L = ML + NL$

Álgebra de ER (II)

- Operador idempotente: el resultado de aplicarlo a dos valores iguales es el mismo valor
  - Propiedad de idempotencia de la unión: $L + L = L$
- Propiedades relativas a las clausuras
  - $(L^*)^* = L^*$
  - $\varnothing^* = \boldsymbol{\varepsilon}$
  - $\boldsymbol{\varepsilon}^* = \boldsymbol{\varepsilon}$
  - $L^+ = LL^* = L^*L$
  - $L^* = L^+ + \boldsymbol{\varepsilon}$
  - $L? = L + \boldsymbol{\varepsilon}$

Álgebra de ER (III)

- Ejemplos: demostrar si las siguientes afirmaciones sobre expresiones regulares son verdaderas o falsas
  - $(R + S)^* = R^* + S^*$
  - $(RS + R)^* R = R(SR + R)^*$
  - $(RS + R)^* RS = (RR^*S)^*$
  - $(R + S)^*S = (R^*S)^*$
  - $S(RS + S)^*R = RR^*S(RR^*S)^*$

Construcción de ER

- Base:
  - Las constantes **ε** y ∅ son ER. L(**ε**) = {**ε**} y L(∅) = ∅
  - Si *a* es un símbolo, **a** es la ER del lenguaje L(**a**) = {*a*}
- Paso inductivo:
  - si *E* y *F* son ER, *E* + *F* es una ER, y $L(E+F) = L(E) \cup L(F)$
  - si *E* y *F* son ER, *EF* es una ER, y *L*(*EF*) = *L*(*E*)*L*(*F*)
  - si *E* es ER, E^* es una ER, y $L(E^*) = (L(E))^*$
  - si *E* es ER, (*E*) es una ER, y *L*((*E*)) = *L*(*E*)

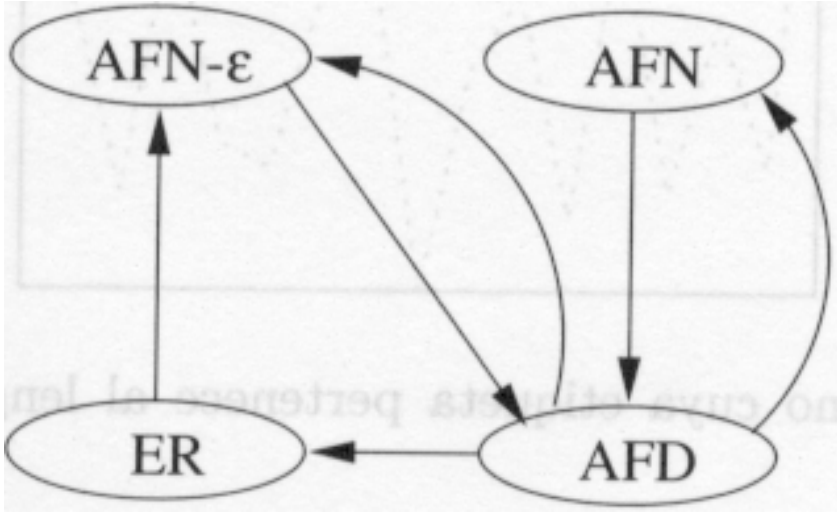
- Ejemplo: escribir la ER para el conjunto de cadenas que constan de ceros y unos alternos

Problemas

1. Escribir las ER para los siguientes lenguajes
   1. El conjunto de cadenas de alfabeto {a, b, c} que contengan al menos una a y una b
   2. El conjunto de cadenas de alfabeto {0, 1} cuyo cuarto símbolo empezando por la izquierda sea un 1.
   3. El conjunto de cadenas de alfabeto {0, 1}, tales que todos los pares de ceros adyacentes aparecen antes que cualquier par de unos adyacentes
2. Describir informalmente el lenguaje representado por la siguiente ER: $(1 + \varepsilon)\ (00^*1)^*0^*$

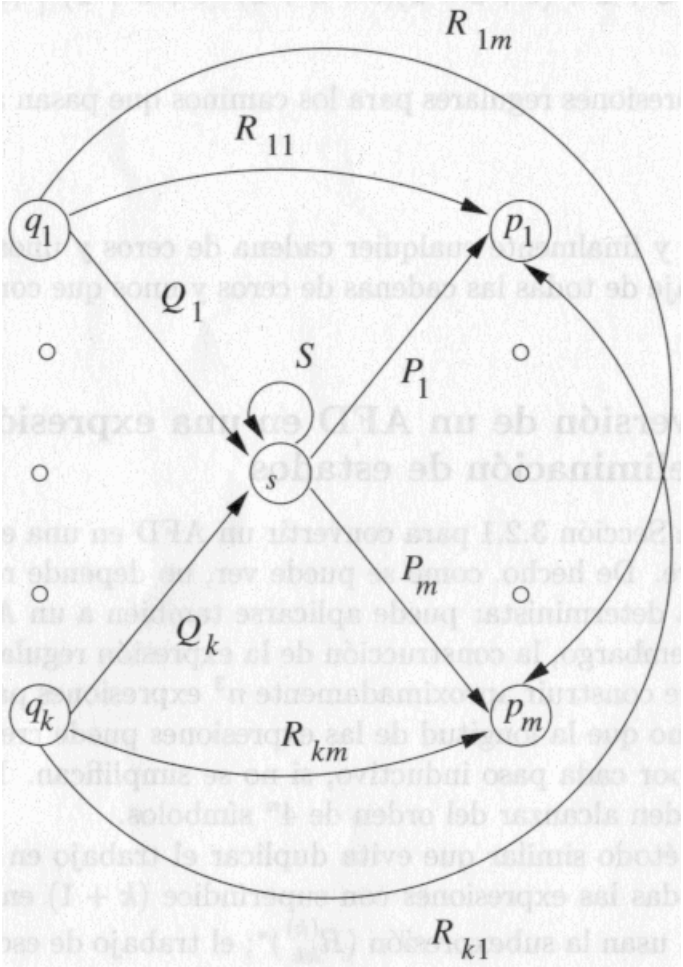
Autómatas finitos y ER

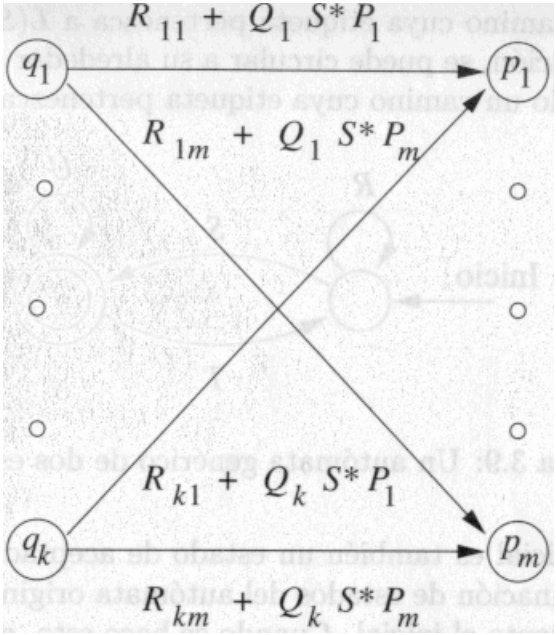
- Las ER definen los lenguajes regulares exactamente igual que los autómatas finitos:
  - Todo lenguaje definido por un AF (usaremos un AFD) también puede definirse mediante una ER
  - Todo lenguaje definido por una ER puede definirse también mediante un AF (usaremos un AFN- ε)



AFD a ER por eliminación de estados

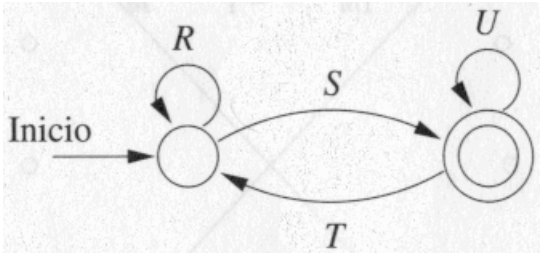
- Eliminación de estados: en los arcos aparecen ER



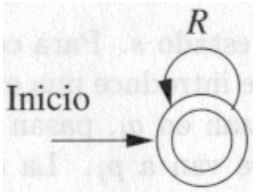


AFD a ER por eliminación de estados (II)

- Para cada estado de aceptación q, se aplica el proceso de reducción. Se eliminan todos los estados excepto q y el estado inicial q_0
- Si $q \neq q_0$, $L = (R + SU^*T)^*SU^*$



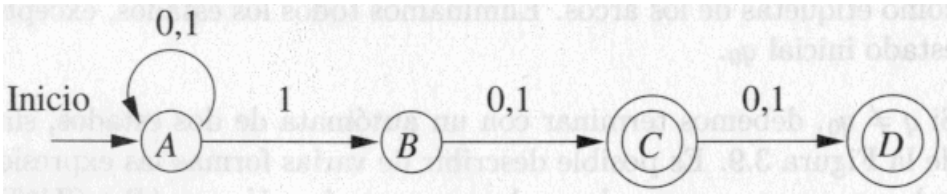
- Si el estado inicial es estado de aceptación, habrá que eliminar todos los estados excepto el inicial: $L = R^*$

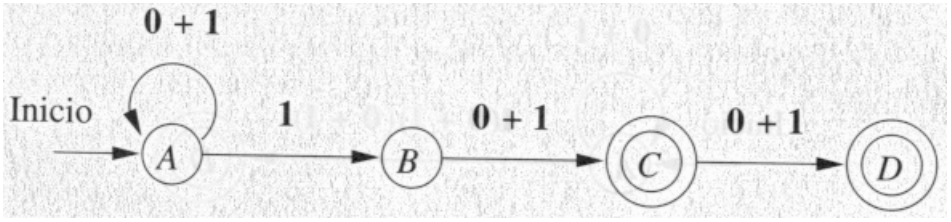


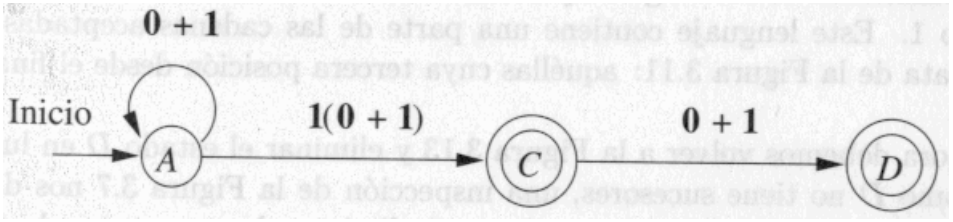
- La ER deseada es la unión de las cadenas obtenidas del autómata para cada estado de aceptación

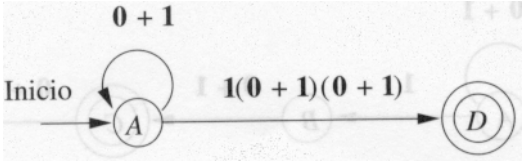
AFD a ER por eliminación de estados (III)

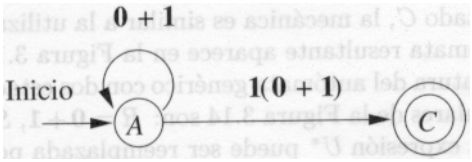
- Ejemplo:







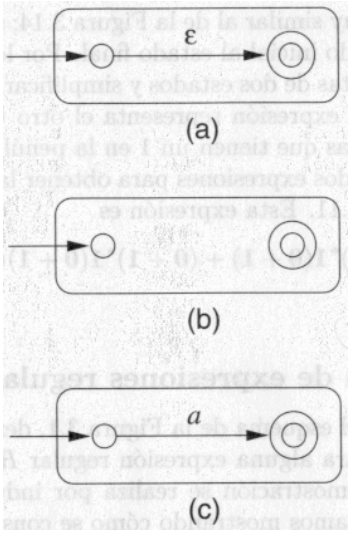




Conversión de ER en autómatas

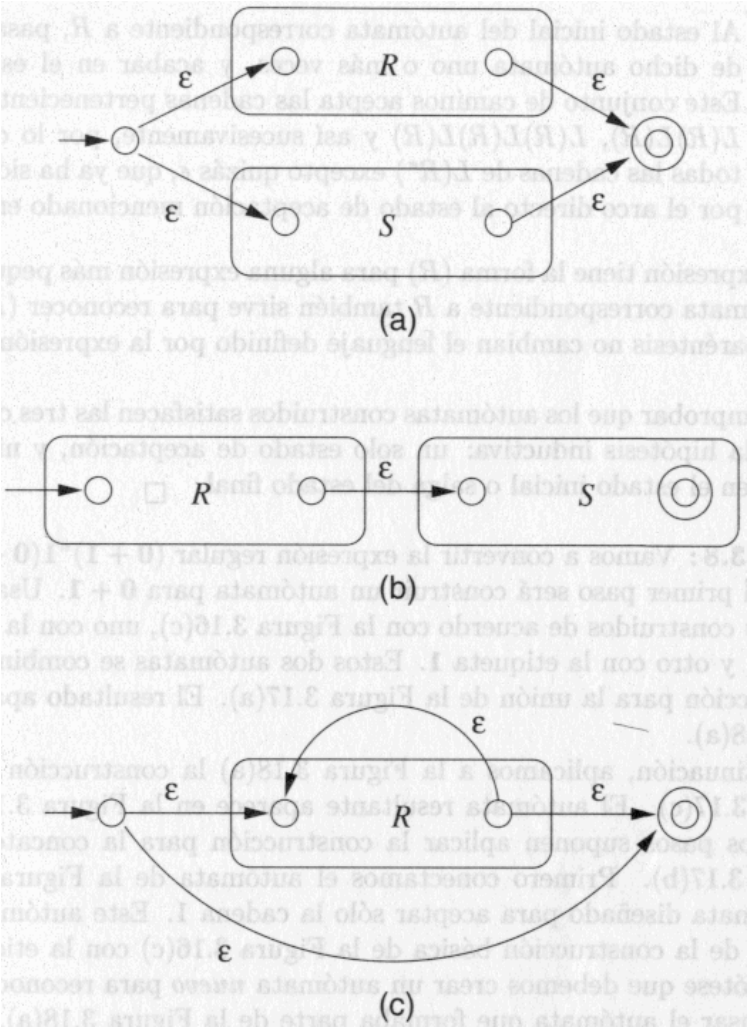
- Teorema: todo lenguaje definido por una ER puede definirse también mediante un AF
- Prueba: sea $L = L(R)$ para la ER R. Se demostrará que $L = L(E)$ para algún AFN- ε E

- Base:



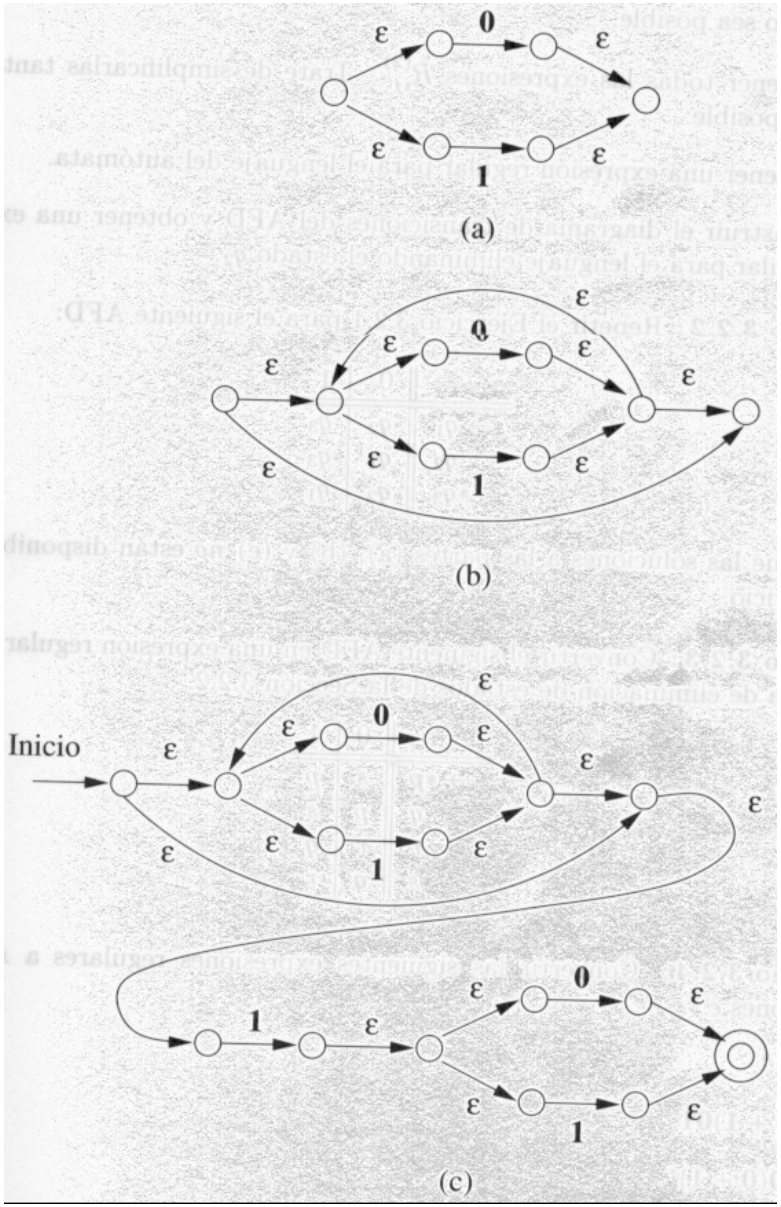
Conversión de ER en autómatas (II)

- Paso inductivo:
  - $R + S$: $L(R) \cup L(S)$
  - RS: $L(R)L(S)$
  - R^*: $L(R^*)$



Conversión de ER en autómatas (III)

- Ejemplo: $(0 + 1)^*1(0 + 1)$



Problemas

1. Dado el siguiente AFD, obtener la ER para el lenguaje del autómata

	0	1
->q_1	q_2	q_1
q_2	q_3	q_1
*q_3	q_3	q_2

Solución: [(1+01) + 00 (0+10) 11]* 00 (0+10)**

2. Convertir el siguiente AFD en una ER

	0	1
->*p	s	p
q	p	s
r	r	q
s	q	r

*Solución: {1 + 0 [(0+10*1)1]* (0+10*1)0}**

3. Convertir las siguientes ER en AFN
   1. 01^*
   2. $(00)^* (0 + 1)^*$

Aplicaciones de las ER

- Búsqueda de patrones de texto mediante ER que dan una "imagen" del patrón que se quiere reconocer
- Aplicaciones
  - analizadores léxicos
  - búsqueda de textos

Búsqueda de patrones en textos

- Búsqueda eficiente de palabras en un gran depósito de texto, como la Web
- La notación de las ER es valiosa para describir patrones de búsqueda interesantes
- Posibilidad de pasar de ER a una implementación eficiente (autómatas)
- Descripción de patrones de texto vagamente definidos: útil emplear ER, ya que es posible modificarlas con poco esfuerzo
- Ejemplo: detección de direcciones de calles en páginas web
  - (Calle|c/|Avenida|Avda\.|Plaza|Pza\.) [A-Z][a-z]*([A-Z][a-z]*)* [0-9]$^{+}$[A-Z]?

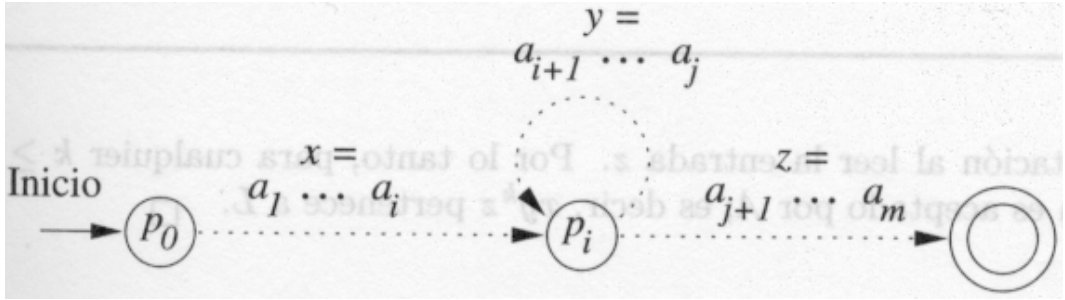
Propiedades de los lenguajes regulares

- Descripción de los lenguajes regulares
  - AFD
  - AFN
  - AFN- ε
  - Expresiones regulares

- No todos los lenguajes son regulares
  - $L_{01} = \{0^n 1^n \mid n \geq 1\}$
    - si fuese regular existiría un AFD con *k* estados que lo reconocería

Lema de bombeo para lenguajes regulares

- Para un lenguaje regular, el cumplimiento del lema de bombeo es una condición necesaria, pero no suficiente
- Teorema: sea *L* un lenguaje regular. Entonces existe una constante *n* (que depende de *L*), tal que, para toda cadena *w* perteneciente a *L*, con $|w| \geq n$, podemos dividir *w* en tres cadenas, *w* = *xyz*, de modo que:
  - $y \neq \varepsilon$
  - $|xy| \leq n$
  - para todo $k \geq 0$, la cadena $xy^k z$ también pertenece a *L*
- Siempre es posible encontrar una cadena no vacía *y*, no demasiado lejos del comienzo de *w*, que se puede "bombear"
- Si se repite *y* cualquier número de veces o se borra (k = 0), la cadena resultante también pertenece al lenguaje *L*



Aplicación del lema del bombeo

1. Elegimos un lenguaje *L* del que queremos demostrar que no es regular
2. El valor de *n* es desconocido, por lo que debemos considerar cualquier posible valor
3. Elegimos *w* (podemos usar *n* como parámetro)
   - Si para un *n* dado no podemos escoger w, *L* será regular
4. Se descompone *w* en *xyz*, sujeto a las restricciones:
   1. $y \neq \varepsilon$
   2. $|xy| \leq n$
5. El lenguaje no será regular si siempre podemos elegir *k* y demostrar que xy^kz no está en *L*
   - Basta encontrar una descomposición de *W* en la que xy^kz esté en *L* para que el lenguaje sea regular

Problemas

1. Demostrar si los siguientes lenguajes son regulares
   1. El que consta de todas las cadenas con el mismo número de 0 y 1 (sin ningún orden particular)
   2. $\{0^p10^p \mid p \geq 1\}$
   3. $\{0^p \mid$ p sea un cuadrado perfecto$\}$
   4. El conjunto vacío
   5. {00, 11}
   6. $(00 + 11)^*$

Propiedades de clausura

- Los lenguajes regulares son cerrados para una serie de operaciones
- Si *L* y *M* son lenguajes regulares, también lo serán:
  - $L \cup M$
  - $L \cap M$
  - *L.M*
  - *L**
  - $\overline{L} = \Sigma^* - L$
  - $L - M$

1. Problema: Utilizando las propiedades de clausura y el hecho de que el lenguaje $L_{0n1n} = \{0^n1^n \mid n \geq 0\}$ no es regular, demostrar que el lenguaje $\{0^i1^j \mid i \neq j\}$ no es regular

Gramáticas regulares

- Otra forma de describir los lenguajes regulares
- Gramáticas regulares:
  - lineales por la derecha
  - lineales por la izquierda
- Una gramática $G = (V, T, P, S)$ es lineal por la derecha si todas sus producciones son de la forma:
    - $A \rightarrow xB$
    - $A \rightarrow x$
  - donde A y B pertenecen a V, y x pertenece a T^*
- Una gramática $G = (V, T, P, S)$ es lineal por la izquierda si todas sus producciones son de la forma:
    - $A \rightarrow Bx$
    - $A \rightarrow x$

Problemas finales

- Dado el autómata finito no determinista AF = ({a, b, c}, {P, Q, R}, *f*, P, {R}), donde *f* está definida en la siguiente tabla de transiciones:

	a	b	c	ε
→ P	P	Q	R	
Q	Q	R		P
* R	R		P	Q

  - Obtener el autómata finito determinista equivalente mínimo.
  - Obtener la expresión regular que representa el lenguaje reconocido por dicho autómata.