

## 3. Lenguajes Regulares

Copyright (c) 2025 Adrián Quiroga Linares Lectura y referencia permitidas;  
reutilización y plagio prohibidos

Las **Expresiones Regulares (ER)** son una forma declarativa de describir lenguajes. Si el Autómata es la máquina que valida, la ER es la "fórmula" que genera las cadenas.

### 3.1 Operadores de las ER

Para leer una ER correctamente, debes conocer la jerarquía de operaciones (como el PEMDAS en matemáticas)

**Los 3 Operadores Básicos:**

1. **Cierre de Kleene / Clausura (\*):** Cero o más repeticiones.


- $L^* = \{\varepsilon, L, LL, LLL, \dots\}$

2. **Concatenación (implícito):** Una cosa detrás de otra.

- $LM$ : Cadenas de  $L$  seguidas de cadenas de  $M$ .

3. **Unión (+ ó |):** Ocurre uno u otro.

- $L + M$ : Cadenas que son de  $L$  o son de  $M$ .

 **Importante: jerarquía de precedencia**

En el examen, si ves  $a + bc^*$ , ¿qué se opera primero?

1. **\* (Lo más fuerte):** Solo afecta a lo que tiene inmediatamente a la izquierda. ( $c^*$ )
2. **Concatenación:** Luego se une. ( $bc^*$ )
3. **Unión (Lo más débil):** Al final se suma. ( $a + (bc^*)$ )

*Usa paréntesis si quieres cambiar esto:  $(a+b)^*.$*

### 3.2 Álgebra de las ER

Estas igualdades sirven para simplificar expresiones complejas

- **Elemento Identidad de la Unión ( $\emptyset$ ):**  $L + \emptyset = L$  (Sumar nada no cambia nada).
- **Elemento Identidad de la Concatenación ( $\varepsilon$ ):**  $L\varepsilon = \varepsilon L = L$  (Concatenar vacío no añade longitud).

- **Elemento Nulo de la Concatenación ( $\emptyset$ ):**  $L\emptyset = \emptyset L = \emptyset$  (Si una parte del camino está rota/vacía, todo el camino se rompe).
- **No conmutativa:**  $ab \neq ba$  (El orden importa).
- **Idempotencia:**  $L + L = L$  (Decir "a ó a" es lo mismo que decir "a").
- **Propiedad del Cierre:**  $\emptyset^* = \varepsilon$  y  $\varepsilon^* = \varepsilon$ .

### 3.3 Conversión de Autómatas Finitos a ER

**Método:** Eliminación de Estados.

La idea es dismantelar el autómata estado por estado hasta que solo quede una "super-flecha" del inicio al final con la Expresión Regular completa.

#### Algoritmo:

1. **Limpieza:** Elimina los **estados sumideros** (o "de muerte") que no llegan a ningún lado.
2. **Limpieza:** Elimina los **estados sumideros** (o "de muerte") que no llegan a ningún lado. Al quitarlo, debes "recablear" las conexiones para no perder información (ver la "Regla del Puente" abajo).
3. Resultado Final:
  - Si el estado inicial es también final: Te quedará 1 solo estado.
  - Si son distintos: Te quedarán 2 estados.
  - **Nota:** Si hay múltiples estados finales, calcula la ER para cada uno por separado (ignorando que los otros son finales) y únelas con un  $+$  al final.

#### Regla del puente

Cuando eliminas un estado intermedio ( $q$ ), cualquier camino que pasaba por él debe convertirse en una flecha directa.

Si tienes:  $A \rightarrow q \rightarrow B$

- Y  $q$  tiene un bucle sobre sí mismo ( $K$ ).
- La nueva flecha directa  $A \rightarrow B$  será:

$$Etiqueta_{nueva} = (Entrada) \cdot (Bucle)^* \cdot (Salida)$$

- Si ya existía una flecha directa de  $A$  a  $B$  con valor *Directo*, se suma:

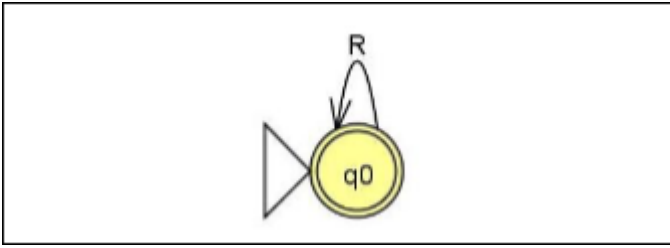
$$Total = Directo + (Entrada \cdot Bucle^* \cdot Salida)$$

#### 1 Solo estado (Inicial = Final)

Si el autómata se reduce a un solo estado con uno o varios bucles.

$$L = R^*$$

**R:** La unión de todas las expresiones de los bucles en el estado ( $r_1 + r_2 + \dots$ ).

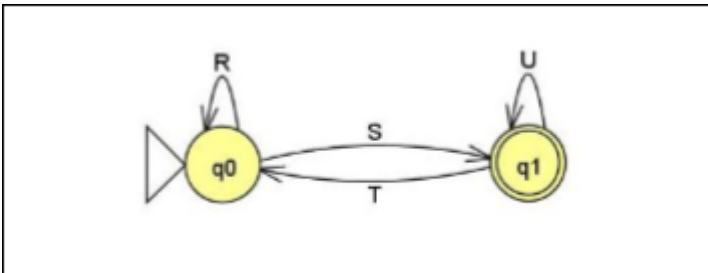


## 2 Estados (Inicial $\neq$ Final)

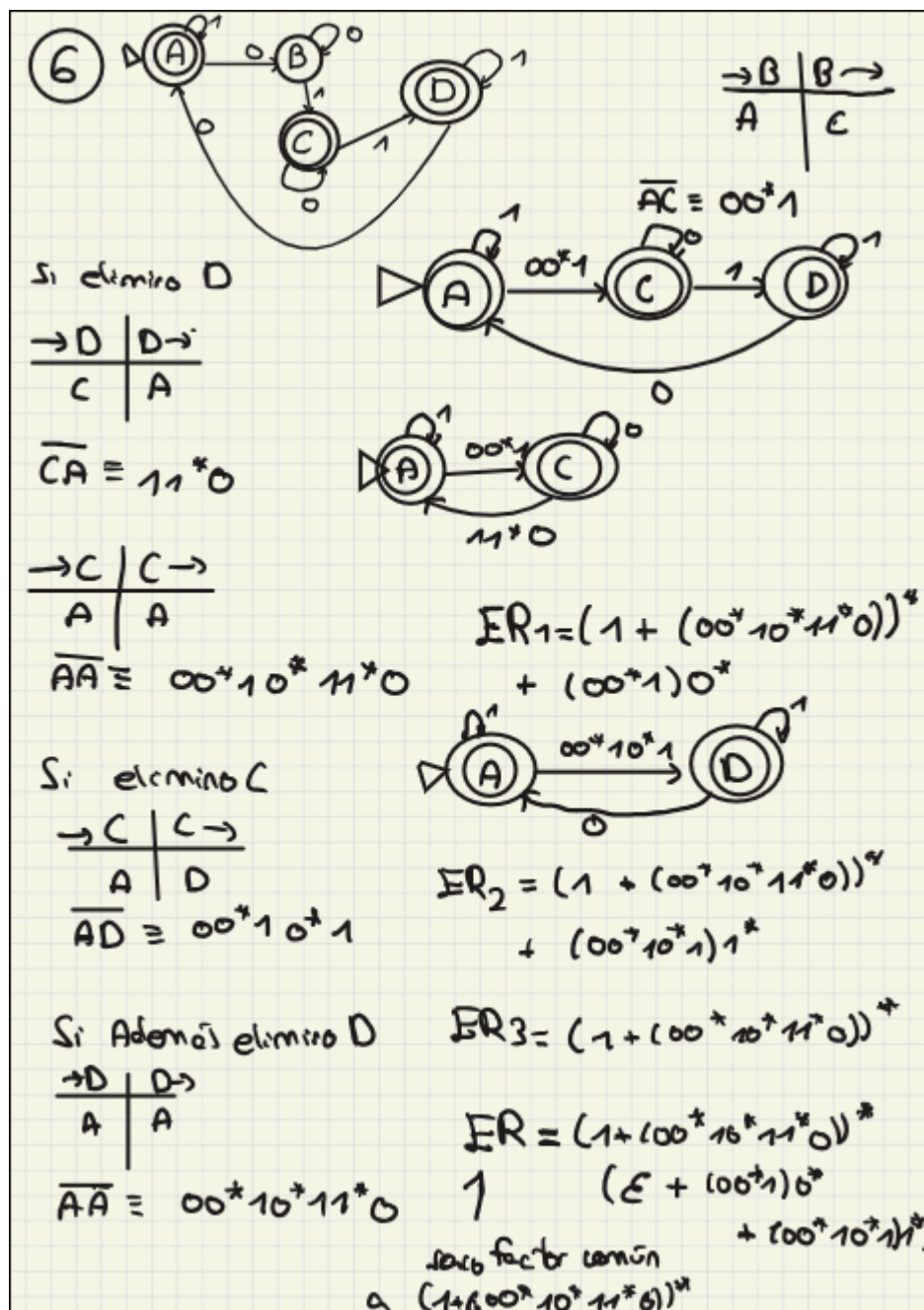
Te queda el estado Inicial ( $q_0$ ) y el Final ( $q_f$ ), con flechas de ida, vuelta y bucles propios.

$$L = (R^* + SU^*T)^* SU^*$$

- $-(R^* + SU^*T)^*$ : Es todo lo que puedes hacer **empezando y acabando en el inicio**.
  - O giras en el inicio ( $R$ ).
  - O vas al final, giras allí y vuelves ( $S \cdot U^* \cdot T$ ).
  - Todo esto repetido las veces que quieras (\*).
- $SU^*$ : Una vez te cansas de dar vueltas en el inicio, **viajas al final ( $S$ )** y puedes quedarte girando allí ( $U^*$ ) para terminar.



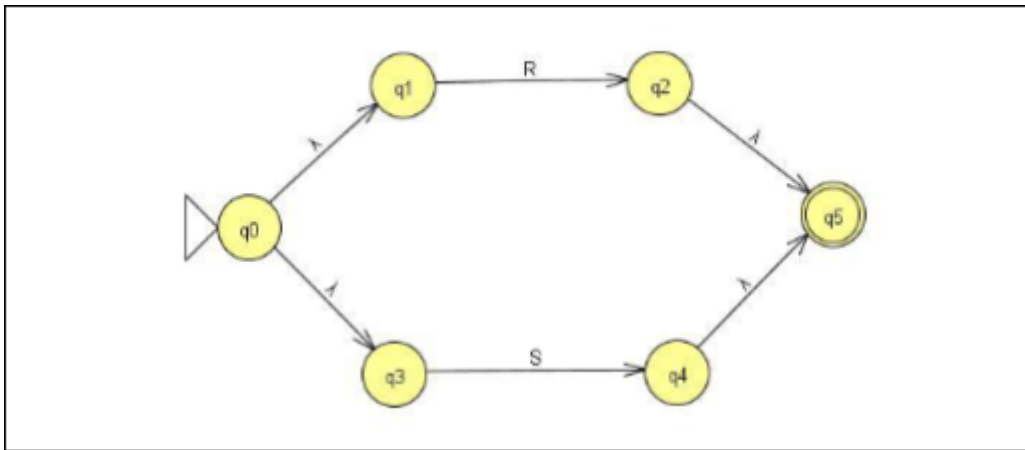
## Ejemplo complejo



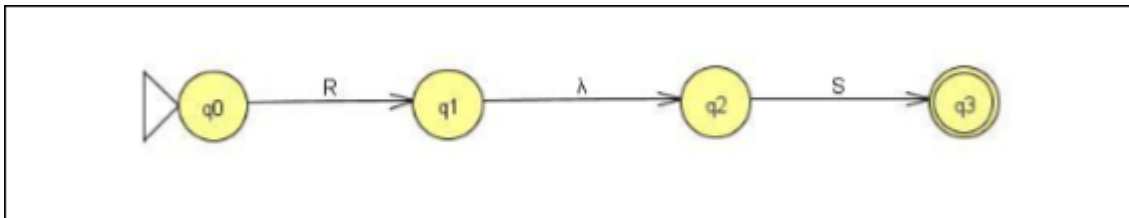
### 3.4 Conversión de ER a Autómatas Finitos

Empleando estas reglas se puede construir un AFD con transiciones epsilon, suelen quedar autómatas gigantescos. Se puede simplificar después o también hay casos donde es obvio el autómatas que reconocen

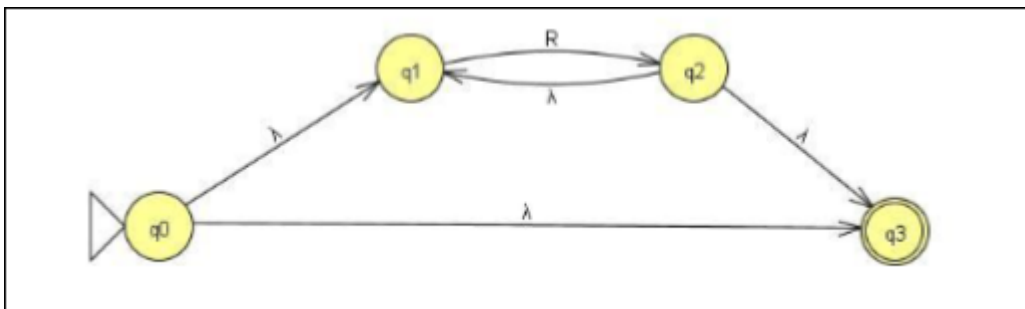
$$R + S : L(R) + L(S)$$



$RS : L(R)(S)$



$R^* : L(R^*)$



**Ejemplo:**

