3. Lenguajes Regulares

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.
   - $L^* = \{\varepsilon, L, LL, LLL, \ldots\}$
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 (ε):** $L\varepsilon = \varepsilon L = L$ (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 desmantelar 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. Al quitarlo, debes "recablear" las conexiones para no perder información (ver la "Regla del Puente" abajo). **Si son finales no los borres**.
2. 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: $A \to q \to B$

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

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

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

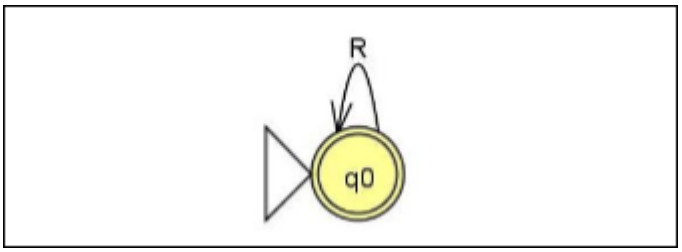
$$Total = Directo + (Entrada \cdot Bucle^* \cdot Salida)$$

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 ($r_1 + r_2 + \ldots$).

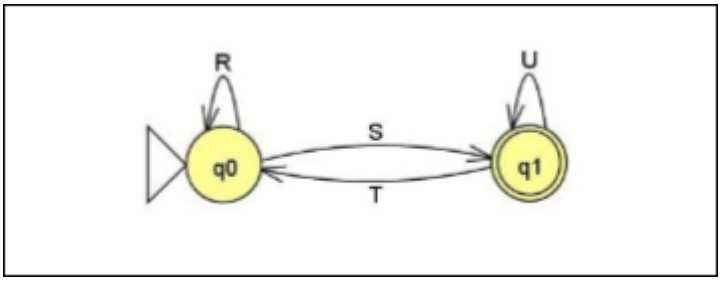


2 Estados (Inicial $\neq$ Final)

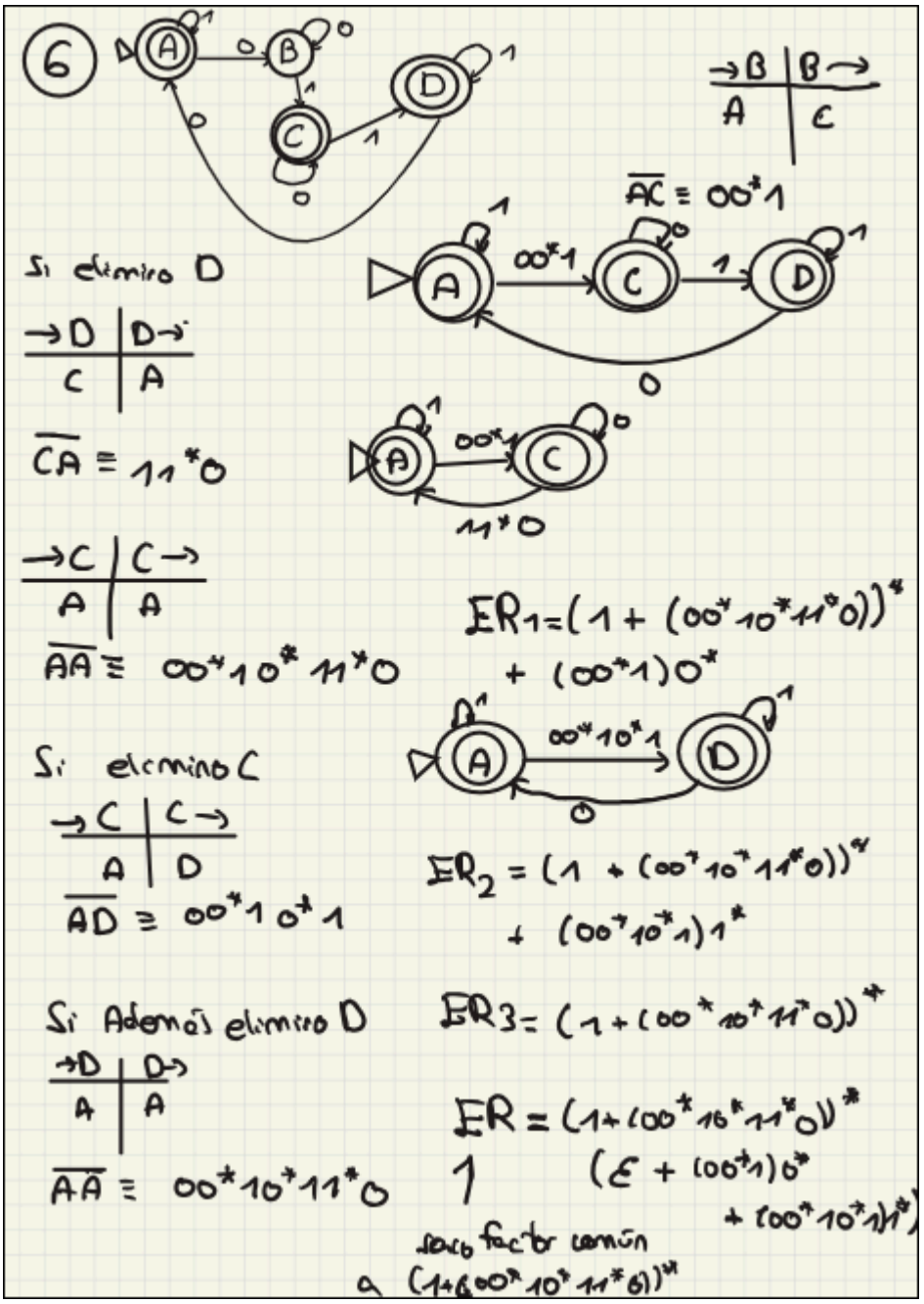
Te queda el estado Inicial (q_0) y el Final (q_f), con flechas de ida, vuelta y bucles propios.

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

- $(R* + SU*T)*$: 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



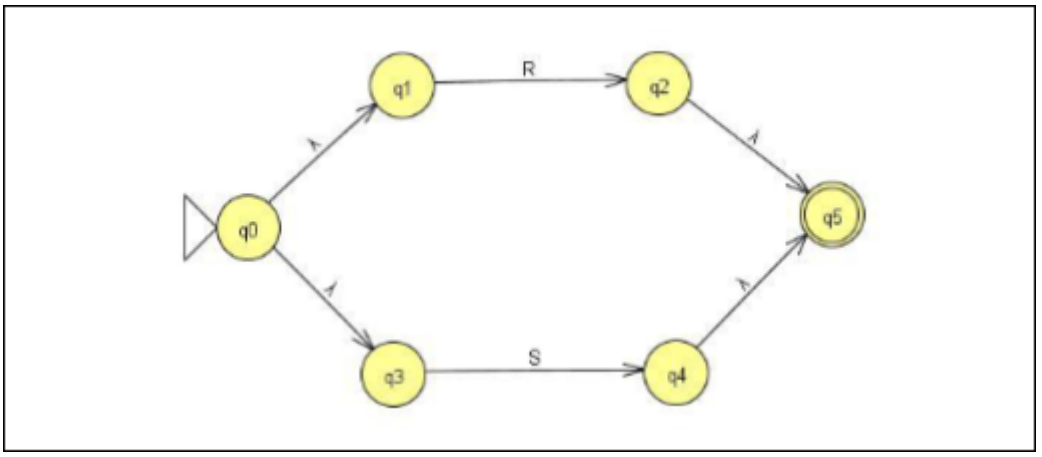
Nota

Cuando hice el ejercicio de arriba se me fue la pinza. En ER_1 donde pone un $+$ es una multiplicación y lo mismo en ER_2. $ER_1 = (1 + (00 * 10 * 11 * 0)) * (00 * 1*)0*$.

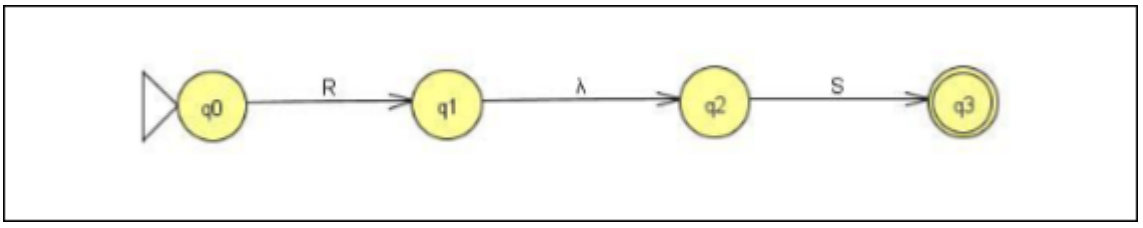
3.4 Conversión de ER a Autómatas Finitos

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 autómata que reconocen

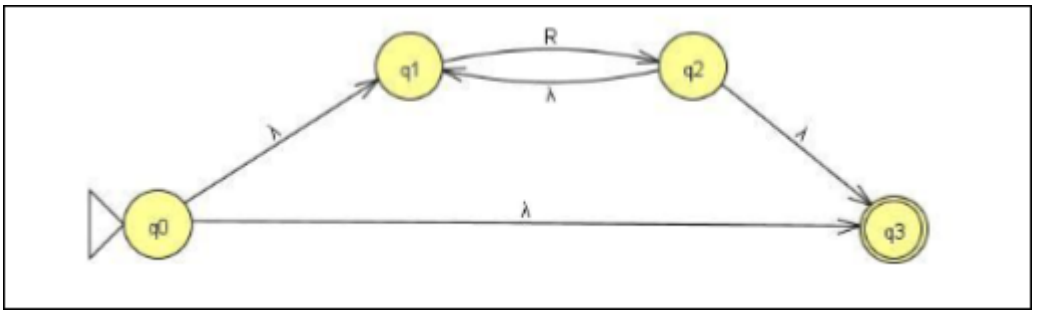
$$R + S : L(R) + L(S)$$



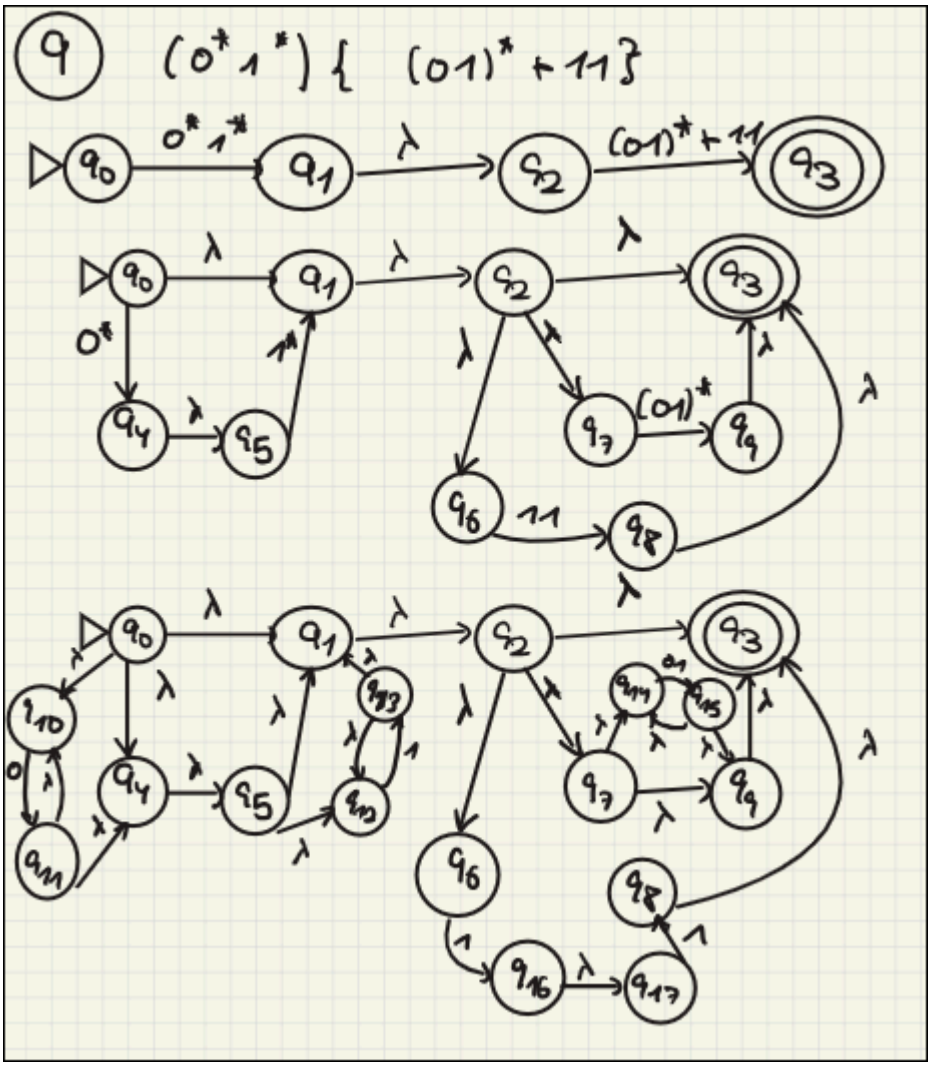
$$RS : L(R)(S)$$



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



Ejemplo:



3.5 Lema del Bombeo para Lenguajes Regulares

> **Nota**
>
> Esto en prácticas no lo dimos, y ns si cae en el final o no pero en anteriores finales lo pregunta. Y claro no se si en la teórica lo dijo porque ir a la teórica nunca fue opción.

Sea L un lenguaje regular. Entonces existe un número entero $p \geq 1$ tal que cualquier cadena w perteneciente a L con longitud $|w| \geq p$ puede dividirse en tres partes, $w = xyz$, cumpliendo las siguientes tres condiciones:

- $|y| > 0$ (o lo que es lo mismo, $y \neq \epsilon$): La parte central no puede estar vacía.
- $|xy| \leq p$: La parte que se repite ocurre dentro de los primeros p caracteres.
- **Para todo $k \geq 0$, la cadena $xy^kz \in L$**: Podemos repetir la parte y tantas veces como queramos (o borrarla con $k = 0$) y la cadena resultante seguirá perteneciendo al lenguaje.

Podemos pensarlo como:

- x **(El prefijo):** El camino desde el inicio hasta antes de entrar al bucle.
- y **(El bucle):** La parte de la cadena que hace que el autómata de una vuelta y regrese al mismo estado. Por eso puedes repetirla (k veces) y el autómata sigue feliz en ese bucle.
- z **(El sufijo):** El camino desde que sales del bucle hasta el estado final.

El Lema del Bombeo no garantiza que el lenguaje sea regular, porque aunque lo cumpla podría no serlo. Lo que es seguro es que si no lo cumple, no es regular.

Ejemplo: $L = \{a^i b^j \mid j = 2i\}$. Eso implica que $N_b = 2 \cdot N_a$ y a mayores se debe de respetar el orden. Normalmente nos darán algo así:

- $x = a^{(n/2)-1}$ (Aporta $\frac{n}{2} - 1$ aes, 0 bes).
- $y = abb$ (Aporta 1 a, 2 bes).
- $z = b^{n-2}$ (Aporta 0 aes, $n - 2$ bes).

Y nos dirán que probemos con varios k si se cumple o no. Si por ejemplo usamos $k = 2$

$$w' = a^{(n/2)-1} \cdot (abb) \cdot (abb) \cdot b^{n-2}$$

Ya podemos apreciar simple vista que va a fallar porque tenemos $abbabb$ lo cual no cumple el orden, por ello no tenemos un lenguaje regular. Pero si nos dicen que $k = 0$ tendríamos

$$w' = x \cdot z$$
$$w' = a^{(n/2)-1} \cdot b^{n-2}$$

Y aprovechamos $N_b = 2 \cdot N_a$ para sustituir:

$$2 \cdot (\frac{n}{2} - 1) = n - 2$$

Vemos que el lema del bombeo no falla, pero esto no significa que sea regular.