Conversión de un AFND a un AFD mediante el método de los subconjuntos

El siguiente algoritmo representa el **procedimiento que transforma un autómata finito no determinista (AFND)** —que puede incluir transiciones múltiples para un mismo símbolo o transiciones vacías (ε)— en un **autómata finito determinista (AFD)**, el cual tiene **una sola transición por símbolo en cada estado**.

La idea fundamental del algoritmo es que **cada estado del nuevo AFD representa un conjunto de estados del AFND original**. Así, el AFD simula de manera determinista todos los posibles movimientos que el AFND podría realizar en paralelo al procesar una cadena de entrada.

Código base del algoritmo

```
s := s0;
c := sgtecar();
while c <> eof do
begin
    s := mueve(s, c);
    c := sgtecar();
end;
if scF then return "SÍ" else return "NO";
```

Este fragmento corresponde al **proceso de reconocimiento de cadenas** en un autómata:

- Se inicia en el estado inicial `s0`.
- Se lee carácter por carácter (`sgtecar()` obtiene el siguiente símbolo).
- En cada paso, se aplica la función `mueve(s, c)` para determinar el nuevo estado alcanzado al consumir el símbolo `c`.
- Al final, si el estado alcanzado pertenece al conjunto de estados finales (`scF`), la cadena es **aceptada**; de lo contrario, **rechazada**.

Construcción del AFD a partir del AFND

```
estaD ← cerr-ε(q0);
while haya un estado T sin marcar en estaD do
    marcar T;
    for cada símbolo de entrada a ∈ Σ do
        U := cerr-ε(mueve(T, a));
        if U ∉ estaD then
            añadir U sin marcar a estaD;
        tranD[T, a] := U;
```

Este bloque describe la **construcción paso a paso del AFD**:

1. **Inicialización:**
   Se calcula la **cerradura ε del estado inicial `q0`**, es decir, todos los estados del AFND a los que se puede llegar desde `q0` usando únicamente transiciones vacías. Este conjunto se convierte en el **primer estado del AFD**, almacenado en `estaD`.

2. **Marcado de estados:**
   Mientras haya un conjunto de estados `T` sin procesar (sin marcar) en `estaD`, se procede a analizarlo.
3. **Transiciones deterministas:**
   Para cada símbolo de entrada `a` del alfabeto Σ:
   - Se calcula `mueve(T, a)`, es decir, todos los estados del AFND a los que se puede llegar desde cualquier estado de `T` consumiendo el símbolo `a`.
   - Luego, se aplica `cerr-ε` sobre ese conjunto para incluir también los estados alcanzables mediante transiciones ε.
   - Si el conjunto resultante `U` **no está** en la lista de estados del AFD (`estaD`), se añade como **nuevo estado sin marcar**.
   - Finalmente, se registra la transición en la tabla `tranD`, de modo que `tranD[T, a] := U`.

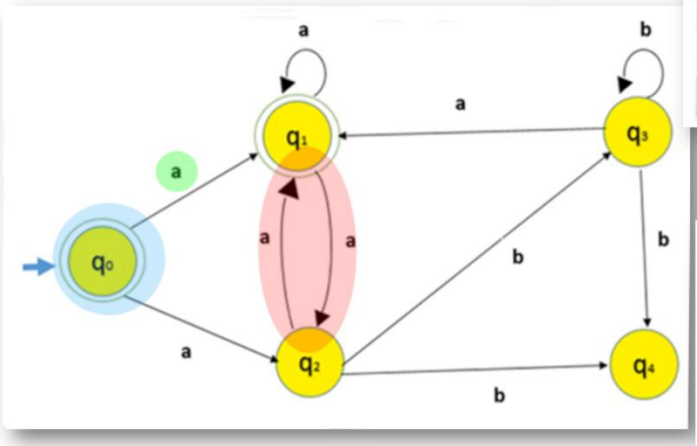
Definiciones de funciones utilizadas

- **`cerr-ε(S)`**:
  Devuelve el conjunto de estados que se pueden alcanzar desde `S` usando solo transiciones ε (sin consumir símbolos de entrada).
- **`mueve(T, a)`**:
  Devuelve el conjunto de estados alcanzables desde cualquiera de los estados de `T` usando el símbolo de entrada `a`.
- **`tranD[T, a] := U`**:
  Define que en el AFD, al estar en el estado `T` y leer el símbolo `a`, se transita al estado `U`.
- **`estaD`**:
  Es la lista que almacena los **conjuntos de estados del AFND**, los cuales forman los **estados del AFD resultante**.

Interpretación general

El algoritmo construye una tabla de transiciones `tranD` que describe completamente el comportamiento del nuevo AFD.
Cada estado del AFD corresponde a **un conjunto de estados del AFND**, y la tabla se completa de manera que el AFD simule **en paralelo** todos los posibles movimientos del AFND para cada símbolo leído.

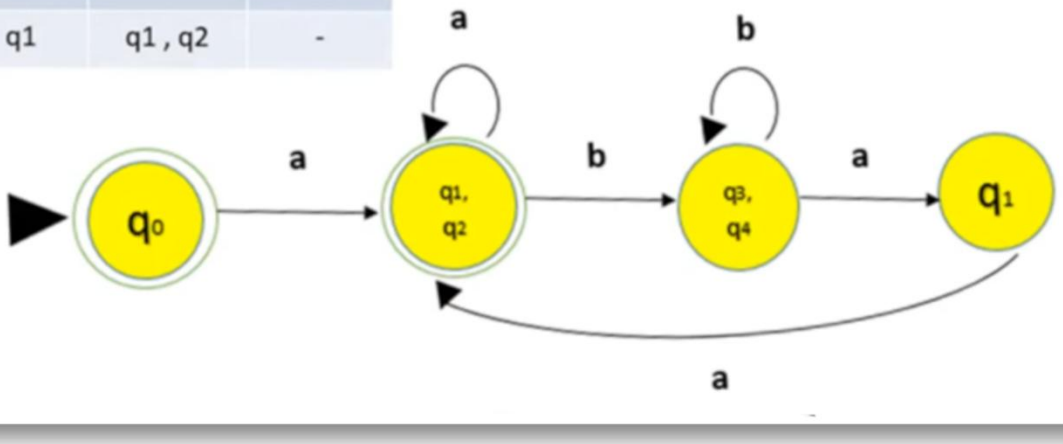
Antes de procesar el primer símbolo de entrada, el AFND puede encontrarse en cualquiera de los estados del conjunto `cerr-ε(s0)`, donde `s0` es el estado inicial.
Desde este conjunto inicial, el algoritmo va generando nuevos conjuntos (estados del AFD) hasta cubrir todas las posibles transiciones del AFND.



	a	b
q_0	q_1 , q_2	-
q_1	q_1 , q_2	-
q_2	q_1	q_3, q_4
q_3	q_1	q_3, q_4
q_4	-	-

	a	b
q_0	q_1 , q_2	-
q_1, q_2	q_1 , q_2	q_3, q_4
q_3, q_4	q_1	q_3, q_4
q_1	q_1, q_2	-

	a	b
q0	q1 , q2	-
q1 , q2	q1 , q2	q3, q4
q3 , q4	q1	q3, q4
q1	q1 , q2	-



Conclusión

El **método de los subconjuntos** o **construcción de subconjuntos** es una técnica fundamental en la teoría de autómatas, ya que permite eliminar el no determinismo y obtener un AFD equivalente a un AFND dado.
Este procedimiento garantiza que ambos autómatas **reconozcan exactamente el mismo lenguaje**, pero el AFD lo hace de forma determinista, lo que facilita su implementación en compiladores y analizadores léxicos.