Construcción de un AFD a partir de una Expresión Regular

La **construcción de un autómata finito determinista (AFD)** a partir de una **expresión regular (ER)** es un proceso sistemático que permite pasar de una descripción simbólica del lenguaje (la ER) a una representación operacional (el AFD), que puede ser implementada directamente para el reconocimiento de cadenas.

1. Expresión regular aumentada

El procedimiento inicia con una **expresión regular aumentada**, denotada como **α$**, donde el símbolo `$` no pertenece al alfabeto original (Σ).
Este símbolo `$` se añade para marcar explícitamente el **final de la cadena**, lo que facilita identificar cuándo el AFD debe aceptar una palabra.

2. Construcción del árbol sintáctico

A partir de **α$**, se construye un **árbol sintáctico** `T` que representa la estructura jerárquica de la expresión regular.
En este árbol:

- Cada **hoja** corresponde a un símbolo terminal (carácter del alfabeto o el símbolo `$`).
- Cada **nodo interno** representa una operación: concatenación, unión o cierre (asterisco).

Las **hojas** del árbol se **numeran** para identificar cada posición dentro de la expresión regular.

3. Cálculo de las funciones base

Una vez construido el árbol, se calculan cuatro funciones fundamentales mediante recorridos sobre `T`:

1. **anulable(n)** → Indica si el subárbol con raíz en `n` puede generar la **cadena vacía (ε)**.
    - Ejemplo: un nodo con `*` siempre es anulable.
2. **primera-pos(n)** → Conjunto de posiciones que **pueden aparecer al inicio** de una cadena generada por el subárbol de `n`.
3. **última-pos(n)** → Conjunto de posiciones que **pueden aparecer al final** de una cadena generada por el subárbol de `n`.
4. **siguiente-pos(i)** → Conjunto de posiciones que **pueden seguir inmediatamente** a la posición `i` en alguna cadena del lenguaje.

Estas funciones se calculan en orden jerárquico, porque **siguiente-pos** depende de los resultados de las tres funciones anteriores.

4. Comprensión del concepto de siguiente-pos

El concepto **siguiente-pos** describe cómo se enlazan los símbolos dentro de las posibles cadenas del lenguaje.
Imagina que cada **hoja** del árbol tiene "vecinos" que pueden venir justo después en una cadena válida.

Por ejemplo, en la secuencia `...cd...`,
si la posición `i` representa la **c**, entonces `siguiente-pos(i)` incluirá la posición de la **d**.

Formalmente, si tenemos un símbolo en una posición, **siguiente-pos(i)** indica **todas las posiciones posibles que pueden aparecer inmediatamente después** en alguna palabra generada por la expresión regular.

5. Reglas para el cálculo de siguiente-pos

La función **siguiente-pos(i)** se obtiene aplicando las siguientes reglas a los nodos del árbol sintáctico:

- **Regla 1: Nodo de concatenación (cat)**
  Si un nodo `n` es una concatenación con hijo izquierdo `c1` e hijo derecho `c2`,
  entonces para cada posición `i ∈ última-pos(c1)`,
  todas las posiciones de `primera-pos(c2)` pertenecen a `siguiente-pos(i)`.

  ∀i ∈ última-pos(c1):\ siguiente-pos(i) ⊇ primera-pos(c2)

- **Regla 2: Nodo de cierre (ast)**
  Si `n` es un nodo con el operador `*`,
  entonces para cada posición `i ∈ última-pos(n)`,
  todas las posiciones de `primera-pos(n)` pertenecen a `siguiente-pos(i)`.

  ∀i ∈ última-pos(n):\ siguiente-pos(i) ⊇ primera-pos(n)

Estas reglas se aplican sistemáticamente a todo el árbol para determinar las posibles transiciones entre símbolos.

6. Interpretación de las funciones

Cada una de las funciones responde a una pregunta clave sobre la estructura de la expresión regular:

- **anulable(n):** ¿El subárbol de `n` puede producir la cadena vacía?
- **primera-pos(n):** ¿Qué símbolos pueden aparecer **al inicio** de la expresión regular α?
- **última-pos(n):** ¿Qué símbolos pueden aparecer **al final** de α?
- **siguiente-pos(i):** Si se ha leído un símbolo en la posición `i`, ¿qué símbolos pueden venir **a continuación**?

7. Relación entre el árbol y el AFD

Después de calcular todas las funciones, se establece una relación entre las **hojas numeradas del árbol** y los **estados del AFD**:

- Cada **estado del AFD** corresponde a un **conjunto de posiciones** (nodos hoja) en el árbol sintáctico.
- Las **transiciones** entre estados se determinan con la función **siguiente-pos**, que indica qué posiciones siguen a otras.

De esta forma, el AFD se construye de manera **directa y sistemática**, a partir de la estructura sintáctica de la expresión regular aumentada.

8. Conclusión

El método de construcción de un AFD a partir de una expresión regular ofrece un enfoque lógico y estructurado que:

- Evita la creación intermedia de un autómata no determinista (AFND).
- Aprovecha la información sintáctica del árbol para deducir las transiciones del AFD.
- Facilita la implementación de analizadores léxicos eficientes en compiladores.

En resumen, numerar las posiciones del árbol, calcular las funciones **anulable**, **primera-pos**, **última-pos** y **siguiente-pos**, y luego construir los estados a partir de estas relaciones, constituye un procedimiento completo para obtener un AFD equivalente a una expresión regular dada.