Ejemplo de paso de AFND a AFD

Partimos de la ER: **(a|b)\*ab?**

El AFND equivalente es:



**Estado de inicio**:

* cierre-λ(0) = {0, 1, 2, 4, 7}, a éste primer estado le llamamos A (es el estado T) y se introduce en estadosD como estado no marcado.
* estadosD = {A} (cuando lo marquemos le pondremos un \*, para saber que ya hemos hecho las operaciones mueve(T, a)

**Comienza la ejecución del algoritmo (1ª iteración)**:

**mientras** haya un estado no marcado T en estadosD **hacer** (en **esta primera iteración A está no marcado)**

marcar T; (en la primera iteración es A el estado que marcamos -A\*-)

**para** cada símbolo de entrada a **hacer** (en nuestro caso los símbolos de la entrada son a y b)

estadosD ={A\*}

transD={Φ} (cadena vacía)

U:= cierre-λ(mueve (T, a))

**Primero calculamos mueve (A, a),** que es el conjunto de estado del AFND que tiene transiciones con la entrada a, desde miembros de A (en este caso el estado 2 y el 7 tiene transición con a, el 2 al 3 y el 7 al 8:

A = {0, 1, 2, 4, 7}

mueve (A, a) = {3, 8} => cierre-λ(mueve (A, a)) = {1, 2, **3**, 4, 6, 7, **8**, 9, 10} = **B**

**NOTA:** El estado 10 aparece en rojo puesto que es el estado final del autómata y por tanto es el estado de aceptación. Por tanto cuando aparezca en un conjunto de estados, como en este caso B, se convierte en estado de aceptación.

El algoritmo dice si U no está en estadosD entonces añadir U como estado no marcado a estadosD, y también transD[T, a] = U

estadosD ={A\*, B}

transD[A, a] = B (**aceptación**)

**Ahora calculamos mueve (A, b) y su cierre- λ**

mueve (A, b) = {5} => cierre-λ(mueve (A, b)) = {1, 2, 4, 5, 6, 7} = **C**

Con esto terminamos la primera iteración y volvemos al bucle" mientras haya un estado no marcado T en estadosD hacer". Como vemos tenemos sin marcar B y C puesto que no hemos pasado ellos..

estadosD ={A\*, B, C}

transD[A, b] = C

**(2ª iteración)**

marcar T --> estadosD = {A\*, B\*, C}

estadosD ={A\*, B\*, C}

transD=[B, a] = B

**para** cada símbolo de entrada a **hacer**

U:= cierre-λ(mueve (T, a))

En nuestro caso hay que hacer U:= cierre-λ(mueve (B, a)) y U:= cierre-λ(mueve (B,b))

B = {1, 2, 3, 4, 6, 7, 8, 9, 10}

mueve(B, a) = {3, 8} => cierre-λ(mueve (B, a)) = {1, 2, **3**, 4, 6, 7, **8**, 9, 10} = **B (es el mismo estado que teníamos)**

estadosD ={A\*, B\*, C, D}

transD=[B, b] = D (**aceptación**)

mueve (B, b) = {5, 10} => cierre-λ(mueve (B, b)) = {1, 2, 4, **5**, 6, **10**} = D

Puesto que tenemos estados no marcados en T (C y D), vamos a por la 3ª iteración

**(3ª iteración)**

estadosD ={A\*, B\*, C\*, D}

marcar T --> estadosD = {A\*, B\*, C\*, D}

**para** cada símbolo de entrada a **hacer**

U:= cierre-λ(mueve (T, a))

En nuestro caso hay que hacer U:= cierre-λ(mueve (C, a)) y U:= cierre-λ(mueve (C,b))

C= {1, 2, 4, 5, 6, 7}

mueve(C, a) = {3, 8} => cierre-λ(mueve (C, a)) = {1, 2, **3**, 4, 5, 6, 7, **8**, 9, 10} = **B (es el mismo estado que teníamos)**

mueve (C, b) = {5} => cierre-λ(mueve (C, b)) = {1, 2, 4, **5**, 6, 7} = C (nos quedamos en el mismo estado)

estadosD ={A\*, B\*, C\*, D}

transD=[C, a] = B (**aceptación**)

transD=[C, b] = C

Como todavía nos queda un estado no marcado en T, (D), hacemos la 4ª iteración:

**(4ª iteración)**

marcar T --> estadosD = {A\*, B\*, C\*, D\*}

**para** cada símbolo de entrada a **hacer**

U:= cierre-λ(mueve (T, a))

En nuestro caso hay que hacer U:= cierre-λ(mueve (D, a)) y U:= cierre-λ(mueve (D,b))

D = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

mueve(D, a) = {3, 8} => cierre-λ(mueve (D, a)) = {1, 2, **3**, 4, 6, 7, **8**, 9, 10} = **B (es el mismo estado que teníamos)**

mueve (D, b) = {5} => cierre-λ(mueve (C, b)) = {1, 2, 4, **5**, 6, 7} = **C** **(es el mismo estado que teníamos)**

estadosD ={A\*, B\*, C\*, D\*}

transD=[D, a] = B (**aceptación**)

transD=[D, b] = C

Todas las iteraciones dan como resultado:

|  |  |  |
| --- | --- | --- |
| ESTADO | ENTRADA | |
| **a** | **b** |
| **A** | B | C |
| **\*B** | B | D |
| **C** | B | C |
| **\*D** | B | C |

Donde B y D son estados de aceptación como indica el asterisco. Nos falta minimizar el AFD que hemos obtenido.

Minimización del AFD

Para realizar la minimización del autómata vamos a utilizar el algoritmo que definen Aho et al (1986) y posteriormente la aplicaremos al ejemplo que estamos desarrollando en este tema. Consiste en lo siguiente:

1. Dividir el conjunto de estados inicial en dos grupos: Los estados de aceptación o finales (F) y los de no aceptación (NF).
2. Aplicar este algoritmo:

**para** cada grupo **hacer**

partir el grupo en subgrupos de tal forma que dos estados s y t

están en el mismo subgrupo si, y sólo si, para todos los símbolos de entrada a, los estados s y t

tienen transiciones en a hacia estados del mismo grupo

/\*en el peor de los casos un estado estará sólo en un subgrupo \*/

sustituir el grupo en Pnueva por el conjunto de todos los subgrupos formados

**fin-para**

1. Si Pnueva := P, hacer Pfinal := P y continuar con el paso 4, si no hacer P := Pnueva y repetir el paso 2.
2. Escoger en cada grupo de la partición un estado representante, que formará parte del AFD mínimo:

* Construir la tabla de transiciones utilizando los estados representantes.
* El estado inicial del AFD mínimo es el representante del grupo que contiene el estado S0 del AFD inicial.
* Los estados finales son los representantes que están en F.

1. Eliminar todos los estados inactivos(estados de no aceptación que tiene transiciones hacía el mismo con todos los símbolos de entrada). También deben eliminarse todos los estados que no sean alcanzables desde el estado inicial.

Representantes de un estado: Cuando hay varios estados que tienen las mismas transiciones para los distintos símbolos de la entrada, estos estados son representados (se eliminan los otros del conjunto) por sólo uno de ellos.

Aplicado a nuestro ejemplo sería:

1. Dividir el conjunto de estados inicial en dos grupos: Los estados de aceptación o finales (B, D) y los de no aceptación (A, C).

En nuestro caso tendremos: F:= {B, D} y NF:= {A, C}

1. Aplicar el algoritmo para obtener Pnueva:
   1. Para los estados de F, con la entrada **b**, B salta a un estado de su grupo,concretamente a D y D sin embargo salta a un estado de otro grupo ©, por lo que podríamos separarlos en: {B} y {D}.
   2. Para los estados de NF (A y C), tanto con la entrada a como con la b, ambos van a los mismos estados, por lo que podemos elegir a uno como representante de ambos estados. Elegimos A, como representatante de A y C por lo que lo sustituimos en la tabla. Éste sería el paso 4 del algoritmo
2. Como ya no se pueden dividir mas, pasamos al paso 4.
3. En esta paso se trata de escoger un representante. Esto haría falta si en alguna partición hubiera mas de un estado, que no es el caso, ya que cada partición se representa si mismo con un sólo estado.
4. Este paso ya no aplica.

Por tanto el AFD mínimo, representado por la tabla de transiciones es el siguiente:

|  |  |  |
| --- | --- | --- |
| ESTADO | ENTRADA | |
| **a** | **b** |
| **A** | B | C |
| **\*B** | B | D |
| **~~C~~** | ~~B~~ | ~~C~~ |
| **\*D** | B | C |

Quedando:

|  |  |  |
| --- | --- | --- |
| ESTADO | ENTRADA | |
| **a** | **b** |
| **A** | B | A |
| **\*B** | B | D |
| **\*D** | B | A |

Dando como resultado final el autómata siguiente:



Para validar el autómata escogemos varios lexemas que cumplan con el patrón de la expresión regular **(a|b)\*ab?**, como por ejemplo: a, aa, aab, aaba, aabab … y veremos que siempre termina el lexema en un estado de aceptación, luego el AFD mínmo que hemos obtenido cumple con la Expresión Regular.