“TECNOLÓGICO NACIONAL DE MEXICO”

INSTITUTO TECNOLÓGICO DE IZTAPALAPA

INTEGRANTES:

ISC-6AM

LENGUAJES Y AUTOMATAS I

M.C. ABIEL TOMÁS PARRA HERNÁNDEZ

SEP 2020 / FEB 2021

ACTIVIDAD SEMANA 10

CUANENEMI CUANALO MARIO ALBERTO

Un autómata finito (AF) o máquina de estado finito es un modelo computacional que realiza cómputos en forma automática sobre una entrada para producir una salida.

Este modelo está conformado por un alfabeto, un conjunto de estados finito, una función de transición, un estado inicial y un conjunto de estados finales. Su funcionamiento se basa en una función de transición, que recibe a partir de un estado inicial una cadena de caracteres pertenecientes al alfabeto (la entrada), y que va leyendo dicha cadena a medida que el autómata se desplaza de un estado a otro, para finalmente detenerse en un estado final o de aceptación, que representa la salida.
La conversión de un AFND-ε en un AFND se basa en el concepto de clausura-ε, que corresponde a una clausura transitiva contextualizada en la teoría de autómatas.

Dado un estado que se llama clausura-ε(q) al conjunto de todos los estados a los que se puede acceder a partir de que, procesando a lo más un único símbolo de la entrada. Puede definirse recursivamente de la siguiente manera:

(Base inductiva) Para todo estado q , $q \in$ clausura-ε(q).

(Inducción) Dados dos estados p y r, si p ∈ clausura-ε(q) y r ∈ δ(p,ε), entonces r ∈ clausura-ε(q).

El algoritmo para eliminar las transiciones vacías es el siguiente:

1. Se calcula la clausura-ε del estado inicial, formándose un conjunto A que corresponderá al estado inicial del nuevo autómata.
2. Para cada símbolo del alfabeto, se verifican los estados alcanzables a partir de algún estado contenido en A , y se calcula la clausura-ε de dichos estados alcanzables. Si dichas clausuras producen nuevos conjuntos distintos de A ,

estos serán nuevos estados a los que se accederá a partir de *A* y del símbolo correspondiente.

3. Se repite lo anterior para cada nuevo conjunto, hasta que no existan transiciones posibles para ningún símbolo del alfabeto.

Ejemplo

En el ejemplo de la figura, se tendrá inicialmente:

clausura-ε(*1*) = {1,2,3,4,6} = *A*

PARA A
Para el símbolo a: 4 va a 5, y clausura-ε(5) = {5,7} = B.
Para el símbolo b: no existen transiciones posibles.
Para B:
Para el símbolo a: no existen transiciones posibles.
Para el símbolo b: 5 va a 6, y clausura-ε(6) = {6} = C.
Para C:
Para el símbolo a: no existen transiciones posibles.
Para el símbolo b: no existen transiciones posibles.

En algunos casos puede ocurrir que al quitar las transiciones épsilon obtengamos directamente un AF D, pues la única razón de no-determinismo era justamente la presencia de dichas transiciones.

Todo AFND (QN, Σ, q0, δN, FN) puede convertirse en un AFD (QD, Σ, q0, δD, FD) equivalente, que mantiene el alfabeto Σ y el estado inicial q0 originales. La conversión implica pasar por un AFD intermedio con estados y transiciones redundantes, que al no ser accesibles a partir del estado inicial, son eliminados para obtener el AFD definitivo.

1. Primero se redefine el conjunto de estados $Q_N = \{ q_0, q_1, ..., q_m\}$ original, como uno conformado por todos los subconjuntos de Q_N. Los nuevos estados finales serán todos aquellos estados que contengan a alguno de los estados finales originales.
2. Posteriormente, se redefine el conjunto de transiciones original, por transiciones del tipo $\delta_D(S, a)$, donde $a \in \Sigma$, y S es la unión de todos los estados q de Q_N para los cuales existía la transición $\delta_N(q, a)$.

3. Por último, se eliminan los **estados inaccesibles** o **inalcanzables** (junto con sus transiciones de salida), es decir, aquellos a los que no se puede acceder a partir del estado inicial. Luego de esta depuración, se obtiene el AFD final.

Minimización de un AFD

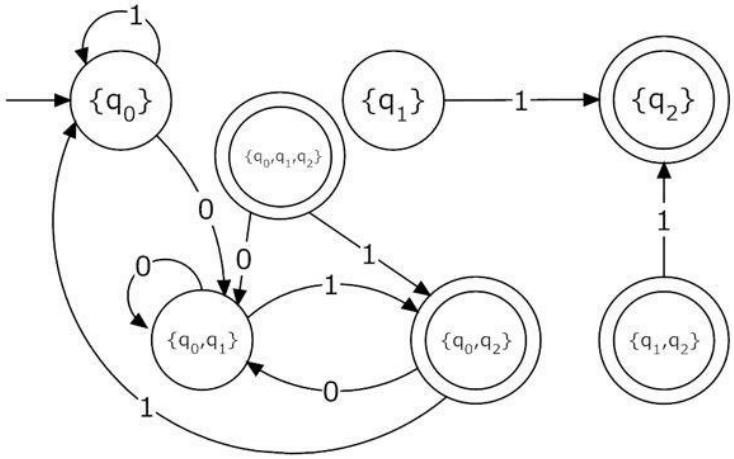
Dos estados de un autómata finito determinista son **estados equivalentes** si al unirse en un solo estado, pueden reconocer el mismo lenguaje regular que si estuviesen separados. Esta unión de estados implica la unión tanto de sus transiciones de entrada como de salida. Si dos estados no son equivalentes, se dice que son **estados distinguibles**. Un estado final con un estado no-final nunca serán equivalentes.

Un AFD está minimizado, si todos sus estados son *distinguibles* y *alcanzables*. Un algoritmo de minimización de AFD es el siguiente:

1. Eliminar los estados inaccesibles del autómata.
2. Construir una tabla con todos los pares (p , q) de estados restantes.
3. Marcar en la tabla aquellas entradas donde un estado es final y el otro es no-final, es decir, aquellos pares de estados que son claramente distinguibles.
4. Para cada par (p, q) y cada símbolo a del alfabeto, tal que $r = \delta(p,a)$ y $s = \delta(q,a)$: 1. Si (r, s) ya ha sido marcado, entonces p y q también son distinguibles, por lo tanto marcar la entrada (p, q).
    2. De lo contrario, colocar (p, q) en una lista asociada a la entrada (r, s).
5. Agrupar a los pares de estados no marcados.

Luego del tercer paso, si la tabla creada queda completamente marcada, entonces el AFD inicial ya era mínimo.

La complejidad computacional del problema de minimizar un AFD es polinomial. De hecho, existen algoritmos más eficientes aún que el mostrado en este artículo (aunque menos intuitivos). Sin embargo, el problema de minimizar un autómata finito no determinista es NP-completo y PSPACE-completo.


{q0}
{q1}
{q2}
{q0,q1,q2}
{q0,q1}
{q0,q2}
{q1,q2}
1
0
1
0
0
1
1
1
0
1