Nombre de la Universidad

Universiada Autómata de Chiapas

Nombre de la Facultad

Facultad de Contaduría y Administración

Nombre de la Licenciatura

Licenciatura en Ingeniería en Desarrollo y Tecnología de Software

Nombre de la Alumna

Zicri Sarai López Hernández

Grado, Grupo y Matricula

6 – N A210297

Nombre de la Actividad

Investigación y ejemplos

Nombre de la Materia

Compiladores

Nombre del Maestro

Luis Gutiérrez Alfaro

Lugar y Fecha

Tuxtla Gutiérrez, Chiapas a 27 de Enero de 2024

Índice

¿Qué es una expresión regular?

Las regex (en inglés, regular expressions) son las unidades de descripción de los lenguajes regulares, que se incluyen en los denominados lenguajes formales. Son un instrumento clave de la informática teórica, la cual, entre otras cosas, establece las bases para el desarrollo y la ejecución de programas informáticos, así como para la construcción del compilador necesario para ello. Es por lo que las expresiones regulares, también denominadas regex y basadas en reglas sintácticas claramente definidas, se utilizan principalmente en el ámbito del desarrollo de software.

Para cada regex existe un denominado autómata finito (también conocido como máquina de estado finito) que acepta el lenguaje especificado por la expresión y que, con ayuda de la construcción de Thompson, se desarrolla a partir de una expresión regular. Por otro lado, para cada autómata finito también hay una expresión regular que describe el lenguaje aceptado por el autómata. Este puede generarse bien con el algoritmo de Kleene o bien con la eliminación de estados.

Tipos de operadores de expresiones regulares.

Comúnmente existen tres operadores de las expresiones regulares: Unión, concatenación y cerradura.

unión

Unión como su nombre dice es la unión de los 2 conjuntos si dicho valor se repite se anota solo una vez. Si L y M son dos lenguajes, su unión se denota por L ∪ M e.g. L = {001,10,111}, M={e,001}, entonces la unión será L ∪ M = {e,10,001,111}.

Concatenación

La concatenación trata de la multiplicación de los conjuntos. El lenguajes se denota como LM o L.M e.g. L={001,10,111}, M={e,001},la concatenación será LM={001,10,111,001001,10001,111001}.

Cerradura

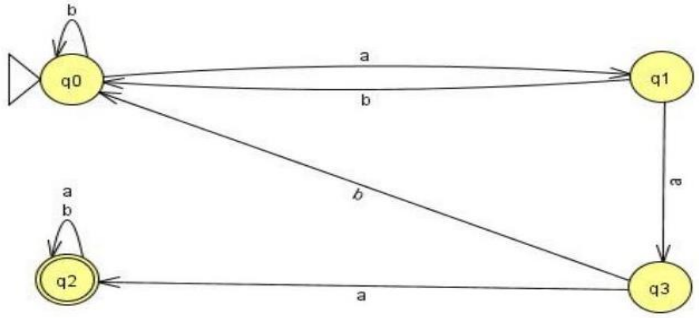
La cerradura (o cerradura de Kleene) de un lenguaje L se denota como L*. Representa el conjunto de cadenas que puede que pueden formarse tomando cualquier número de cadenas de L, posiblemente con repeticiones y concatenando todas ellas e.g. Si L = {0, 1}, L∗ son todas las cadenas con 0's y 1's. Si L = {0, 11}, entonces L∗ son todas las cadenas de 0's y 1's tal que los 1's están en pareja. Para calcular L∗ se debe calcular L i para cada i y tomar la unión de todos estos lenguajes. L^i tiene 2^i elementos. Aunque cada L^i es finito, la unión del número de términos de L^i es en general un conjunto infinito e.g. ∅∗ = {e} o ∅ 0 = {e}. Generalizando, para todo i mayor o igual que uno, ∅^i es el conjunto vació, no se puede seleccionar ninguna cadena del conjunto vacío.

Proceso de conversión de DFA a expresiones regulares.

Básicamente este método consiste en seleccionar tres estados: q_Y, el cual no deberá ser ni el estado inicial, ni ninguno de los estados finales o de aceptación, también se deberá seleccionar un estado qx y qy, de manera que qx pueda llegar (por medio de transiciones) a qy utilizando a q_Y como estado intermedio entre estos. Después de haber seleccionado estos estados, se debe proceder a eliminar el estado q_Y, haciendo una transición que vaya de qx a qy y que por medio de la concatenación de las transiciones que llegan de qx a q_Y y salen de q_Y a qy (incluyendo las que hacen un bucle en q_Y). En caso de que ya exista una transición que va de qx a qy, se hace la unión de la Expresión Regular de dicha transición con la Expresión Regular de la nueva transición antes creada. Esto se repite hasta que solo existan estados iniciales y finales en el DFA. Luego de tener la máquina de esta forma se debe generar la Expresión Regular a partir de ella.

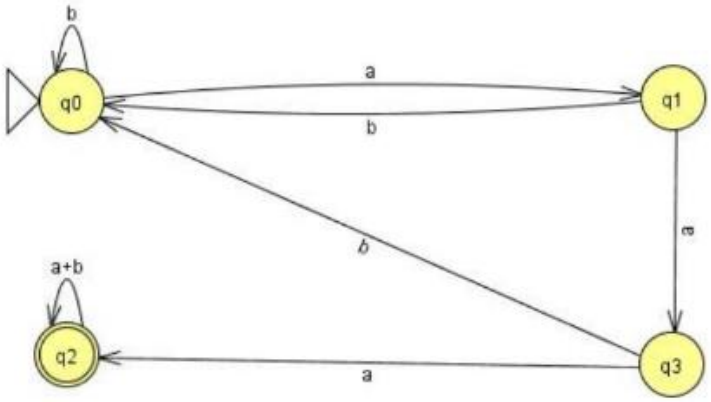
Ejemplo

Haremos la conversión del siguiente DFA a una Expresión



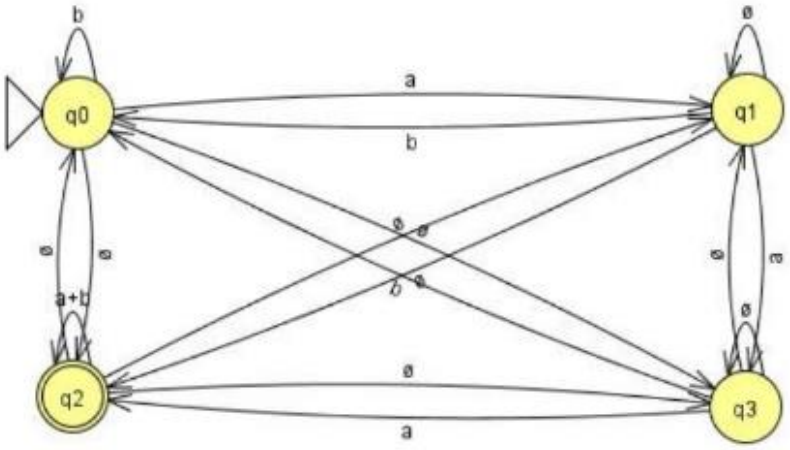
PASO 1: Por cada transición Qi^3 que pueda ser recorrida con múltiples símbolos, se hará una transición Qj (siendo esta una transición que contiene una Expresión Regular) que contenga los símbolos de dicha transición Qi representados como una Expresión Regular, específicamente como una unión.

APLICACIÓN: Como podemos observar, la transición que hace un bucle en q^2 es la única transición que tiene múltiples símbolos con la cual puede ser transitada, por lo tanto la representaremos como una unión de la siguiente manera: a + b. Nos resulta en:



PASO 2: Por cada estado qi, se debe verificar si hay una transición Qj que llegue a cada estado qn (donde qn = qi) de la máquina. En caso de no existir esta transición se deberá agregar una transición que va desde qi hasta qn con el valor ∅.

APLICACIÓN: Al aplicar el paso 2 a nuestro DFA, podemos ver que no hay transición de q0 a q2, tampoco existe un bucle en q1, tampoco hay transición de q3 a q2 etc. Por lo tanto agregaremos todas las transiciones que hacen falta para conectar cada estado con el resto de estados de la máquina. Estos estados tendrán el símbolo ø. Por lo tanto nuestro DFA resulta en:



PASO 3: Seleccionar un estado qr, talque qr NO sea un estado inicial y/o final. Luego, por cada estado qx se selecciona un camino, pasando por qr, hacia cada estado qy del DFA, talque qx ≠qr y qy ≠qr. Ahora se crea una transición Qj que tenga como Expresión Regular el símbolo (o Expresión Regular) de la transición que va de qx a qr concatenado con el símbolo de la transición que va de qr a qy. Al bucle que se hace en qr se le aplicará la operación de clausura (o clausura Kleene) y se concatenará con la Expresión Regular antes encontrada. A esta nueva transición Qj se le aplica una operación de unión con el símbolo de la transición que va de qx a qy. La transición Qj deberá quedar de la forma TiS*Tj+Tk. Esta nueva transición Qj transitará del estado qx al estado qy

APLICACIÓN: Ahora bien, seleccionaremos como estado qr a q1. Haciendo la concatenación da cada transición desde todos los estados qx hacia todos los estados qy usando a qr como intermediario, nos resulta las siguientes Expresiones Regulares:

q_x	q_y	Q_j
0	0	ab
0	2	$a\emptyset$
0	3	aa
2	0	$b\emptyset$
2	2	$b\emptyset$
2	3	$a\emptyset$
3	0	$b\emptyset$
3	2	$b\emptyset$
3	3	$a\emptyset$

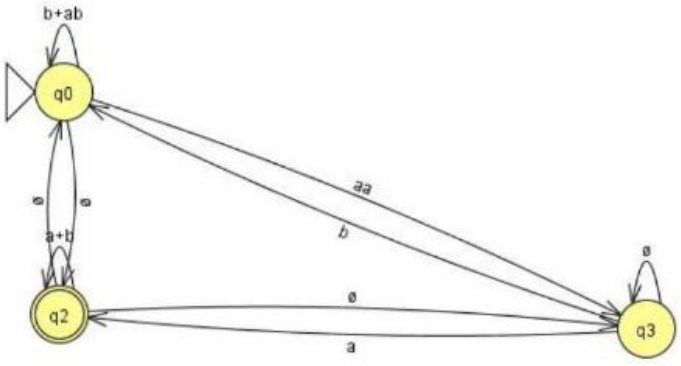
Ahora le aplicaremos la operación de clausura al bucle de qr y haremos la concatenación con el Qj que ya encontramos. Resulta en:

q_x	q_y	Q_j
0	0	$a\emptyset^*b$
0	2	$a\emptyset^*\emptyset$
0	3	$a\emptyset^*a$
2	0	$b\emptyset^*\emptyset$
2	2	$b\emptyset^*\emptyset$
2	3	$a\emptyset^*\emptyset$
3	0	$b\emptyset^*\emptyset$
3	2	$b\emptyset^*\emptyset$
3	3	$a\emptyset^*\emptyset$

El siguiente paso es hacer la unión del Qj que ya tenemos con el símbolo de la transición que va de qx z qy directamente. También agregaremos una columna con la Expresión Regular ya simplificada, por lo tanto obtenemos:

q_x	q_y	Q_j	Simplificación
0	0	$a\emptyset^*b + b$	$\boldsymbol{ab + b}$
0	2	$a\emptyset^*\emptyset + \emptyset$	$\emptyset$
0	3	$a\emptyset^*a + \emptyset$	$\boldsymbol{aa}$
2	0	$b\emptyset^*\emptyset + \emptyset$	$\emptyset$
2	2	$b\emptyset^*\emptyset + a + b$	$\boldsymbol{a + b}$
2	3	$a\emptyset^*\emptyset + \emptyset$	$\emptyset$
3	0	$b\emptyset^*\emptyset + b$	b
3	2	$b\emptyset^*\emptyset + a$	a
3	3	$a\emptyset^*\emptyset + \emptyset$	$\emptyset$

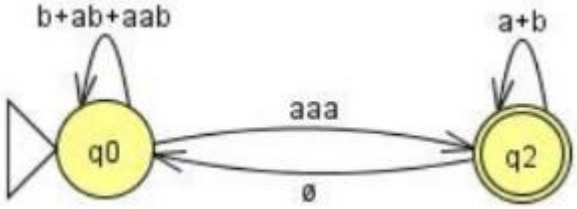
Excelente! Hemos logrado eliminar el estado q1 de nuestro DFA. Ahora se deben seguir los mismos pasos hasta tener un DFA que solo contenga el estado inicial y los finales (en este caso 1 q0 y q2) . El DFA nos queda de la siguiente manera:



Ahora que ya hemos comprendido los pasos esenciales de la eliminación de estados, presentaremos la tabla para eliminar el estado qa:

q_x	q_y	Q_j	Simplificación
0	0	$aa\emptyset^*b + ab + b$	$aab + ab + b$
0	2	$aa\emptyset^*a + \emptyset$	aaa
2	0	$\emptyset\emptyset^*a + \emptyset$	$\emptyset$
2	2	$\emptyset\emptyset^*a + b + a$	$b + a$

Hemos terminado de eliminar los estados no iníciales y no finales de nuestro DFA. Aplicando todas las Expresiones Regulares de la tabla anterior a nuestro DFA, nos quedaría así:



PASO 4: Si tenemos un estado inicial qx y un estado final qy donde qx≠ qy, se debería generar una Expresión Regular a partir de este DFA de la forma (+SU*T)*SU*, donde R es un bucle en qx, S es el camino que va de qx a qy, U es un bucle en q2y T es el camino que va de qy a qx.

APLICACIÓN: Primero identificamos las Expresiones Regulares R, S, U y T. Tenemos que R=b+aab+a,S=aaa,U=a+b,yT=ø. Ahora que ya tenemos los valores, procedemos a hacer la Expresión Regular final:

$$\boldsymbol{ER} = (\boldsymbol{b} + \boldsymbol{ab} + \boldsymbol{aab} + (\boldsymbol{aaa})(\boldsymbol{a} + \boldsymbol{b})^*\emptyset)^*(\boldsymbol{aaa})(\boldsymbol{a} + \boldsymbol{b})^*$$

$$= (\boldsymbol{b} + \boldsymbol{ab} + \boldsymbol{aab})^*(\boldsymbol{aaa})(\boldsymbol{a} + \boldsymbol{b})^*$$

leyes algebraicas de expresiones regulares.

Existen un conjunto de leyes algebraicas que se pueden utilizar para las expresiones regulares:

• Ley conmutativa para la unión: L + M = M + L

• Ley asociativa para la unión: (L + M) + N = L + (M + N)

• Ley asociativa para la concatenación: ´ (LM)N = L(MN)

NOTA: La concatenación no es conmutativa, es decir LM ≠ ML

Leyes distribuidas

• Como la concatenación no es conmutativa, tenemos dos formas de la ley distributiva para la concatenación

• Ley Distributiva Izquierda para la concatenación sobre unión: L(M + N) =LM+ LN

• Ley Distributiva Derecha para la concatenación sobre unión: (M + N)L = ML + NL

Ley de Idempotencia

• Se dice que un operador es idempotente (idempotent)si el resultado de aplicarlo a dos argumentos con el

mismo valor es el mismo valor

• En general la suma no es idempotente: X + X ≠ X (aunque para algunos valores sí aplica como 0 + 0 = 0)

• En general la multiplicación tampoco es idempotente: X × X ≠ X

• La unión e intersección son ejemplos comunes de operadores idempotentes. Ley idempotente para la unión + L = L

Leyes que involucran la cerradura

• (L∗)∗ = L∗ (Idempotencia para la cerradura)

• ∅∗ = e

• e∗ = e

• L^+ = LL∗ = L∗L,L^+ se define como L + LL + LLL + . . .

• L∗ = e + L + LL + LLL + . . .

• LL∗ = Le + LL + LLL + LLLL + . . .

• L∗ = L^+ + e

• L? = e + L

Bibliografías

Dfa A Expresion Regular. (n.d.). Doku.Pub. Retrieved January 27, 2024, from https://doku.pub/documents/dfa-a-expresion-regular-oj0v6ny42pqx

Regex o expresiones regulares: la manera más sencilla de describir secuencias de caracteres. (2019, December 30). IONOS Digital Guide; IONOS. https://www.ionos.mx/digitalguide/paginas-web/creacion-de-paginas-web/regex/

Regulares, 1. Expresiones. (n.d.). *Propedeutico: Teor´ıa de Aut´omatas y Lenguajes Formales Expresiones regulares y lenguajes*. Inaoep.Mx. Retrieved January 27, 2024, from https://posgrados.inaoep.mx/archivos/PosCsComputacionales/Curso_Propedeutico/Automatas/03_Automatas_ExpresionesRegularesLenguajes/CAPTUL1.PDF#:~:text=Comunmente%20existen%20tres%20operadores%20de%20las,expresiones%20regulares%3A%20Union%2C%20concatenacion%20y%20cerradura.

(N.d.). Inaoep.Mx. Retrieved January 27, 2024, from https://ccc.inaoep.mx/ingreso/automatas/expresionesRegulares.pdf