UNIVERSIDAD AUTONOMA DE CHIAPAS

FACULTAD DE CONTADURIA

LIC. EN INGENIERIA Y DESARROLLO DE SOFTWARE

MATERIA

Compiladores

DOCENTE

Dr. Luis Alfaro Gutiérrez

TRABAJO

Define los siguientes conceptos y realizar los ejercicios. - Actividad I, II.

ESTUDIANTE

KEVIN MARTIN MORALES HERRERA

6to semestre grupo “O”

TUXTLA GUTIERREZ, CHIAPAS. 28 de enero de 2024

indicce

Actividad I.- Investigación y Ejemplos.

Expresiones regulares

Las expresiones regulares son un equivalente algebraico para un autómata. Utilizado en muchos lugares como un lenguaje para describir patrones en texto que son sencillos pero muy ´útiles. Pueden definir exactamente los mismos lenguajes que los autómatas pueden describir: Lenguajes regulares. Además, ofrecen algo que los autómatas no: Manera declarativa de expresar las cadenas que queremos aceptar. Sirven como lenguaje de entrada a muchos sistemas que procesan cadenas tales como:
• Comandos de búsqueda, e.g., grep de UNIX
• Sistemas de formateo de texto: Usan notación de tipo expresión regular para describir patrones • Convierte la expresión regular a un DFA o un NFA y simula el autómata en el archivo de búsqueda • Generadores de analizadores-léxicos. Como Lex o Flex.
• Los analizadores léxicos son parte de un compilador. Dividen el programa fuente en unidades lógicas (tokens). Tokens como while, números, signos (+, -,<,etc).

Comunmente existen tres operadores de las expresiones regulares: Uni´on, concatenaci´on y cerradura. Si L y M son dos lenguajes, su uni´on se denota por L ∪ M e.g. L = {001,10,111}, M ={ ,001}, entonces la uni´on ser´a L ∪ M = { ,10,001,111}. La concatenaci´on de lenguajes se denota como LM o L.M e.g. L = {001,10,111}, M ={ ,001}, entonces la concatenaci´on ser´a LM ={001,10,111,001001,10001,111001}. Finalmente, 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 n´umero 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 est´an en pareja. Para calcular $L*$ se debe calcular L i para cada i y tomar la uni´on de todos estos lenguajes. L i tiene 2i elementos. Aunque cada L i es finito, la uni´on del n´umero de terminos de L i es en general un

conjunto infinito e.g. ∅ ∗ = { } o ∅ 0 = { }. Generalizando, para todo i mayor o igual que uno, ∅ i es el conjunto vac´ıo, no se puede seleccionar ninguna cadena del conjunto vac´ıo

Sintaxis de las expresiones regulares

carácter

En una expresión regular, un *carácter* es un carácter XML normal que no es un metacarácter.

metacaracteres

Los metacaracteres son caracteres de control en las expresiones regulares. Los metacaracteres de expresiones regulares que están soportados actualmente son:

barra inclinada invertida (\)

Inicia un escape de clase de carácter. Un escape de clase de carácter indica que el metacarácter siguiente debe utilizarse como carácter, en lugar de un metacarácter.

punto (.)

Coincide con cualquier carácter individual, excepto con el carácter de nueva línea (\n).

signo de intercalación (^)

Si el carácter de intercalación aparece fuera de una clase de carácter, los caracteres que le siguen coinciden con el inicio de la serie de entrada o, para series de entrada de varias líneas, con el inicio de una línea. Una serie de entrada se considera como una serie de entrada de varias líneas si la función que utiliza la serie de entrada incluye el distintivo *m*.

Si el carácter de intercalación aparece como primer carácter en una clase de carácter, el carácter de intercalación actúa como un signo de negación. Se produce una coincidencia si ninguno de los caracteres del grupo de caracteres aparece en la serie que se compara con la expresión regular.

signo de dólar ($)

Coincide con el final de la serie de entrada o, para series de entrada de varias líneas, con el final de una línea. Una serie de entrada se considera como una serie de entrada de varias líneas si la función que utiliza la serie de entrada incluye el distintivo *m*.

signo de interrogación (?)

Coincide con el carácter o grupo de caracteres anteriores de la expresión regular cero o una vez.

asterisco (*)

Coincide con el carácter o grupo de caracteres anteriores de la expresión regular de cero o más veces.

signo Más (+)

Coincide con el carácter o grupo de caracteres anteriores de la expresión regular una o más veces.

barra vertical (|)

Coincide con el carácter (o grupo de caracteres) anterior o el carácter (o grupo de caracteres) siguiente.

corchete de apertura ([) y corchete de cierre (])

Los corchetes de apertura y cierre y el grupo de caracteres que incluyen definen una clase de carácter. Por ejemplo, la clase de carácter [aeiou] coincide con cualquier vocal. Las clases de carácter también admiten rangos de caracteres. Por ejemplo:

- [a-z] significa cualquier letra minúscula.
- [a-p] significa cualquier letra minúscula de la a a la p.
- [0-9] significa cualquier dígito único.

paréntesis de apertura (() y paréntesis de cierre ())

Un paréntesis de apertura y uno de cierre indican una agrupación de algunos caracteres dentro de una expresión regular. A continuación, puede aplicar un operador, como por ejemplo un operador de repetición, a todo el grupo.

La llave de apertura ({) y la llave de cierre (}) también son metacaracteres, pero actualmente no están soportados.

escape-clase-carácter

Un escape de clase de carácter especifica que determinados caracteres especiales deben tratarse como caracteres, en lugar de realizar alguna función. Un escape de clase de carácter consta de una barra inclinada invertida (\), seguida de un único metacarácter, carácter de nueva línea, carácter de retorno o carácter de tabulación. La tabla siguiente lista los escapes de clase de carácter.

Conversión de DFA a Expreciones regulares

METODO DE ELIMINACION

Básicamente este método consiste en seleccionar tres estados: qR 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 qr, como estado intermedio entre estos. Después de haber seleccionado estos estados, se debe proceder a eliminar el estado qr, haciendo una transición que vaya de q a qy y que por medio de la concatenación de las transiciones que llegan de qx a qy salen de q, a qy (incluyendo las que hacen un bucle en q,). 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 de método eliminacion

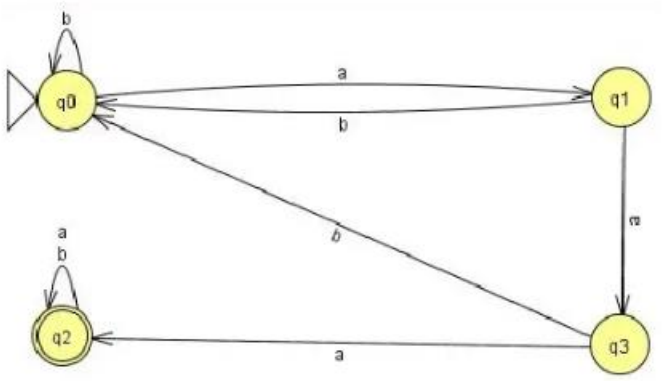

ILUSTRACIÓN 1: DFA ORIGINAL

PASO 1: Por cada transición Q_i[3] que pueda ser recorrida con múltiples símbolos, se hará una transición Q_j(siendo esta una transición que contiene una Expresión Regular)[4] que contenga los símbolos de dicha transición Q_i 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:

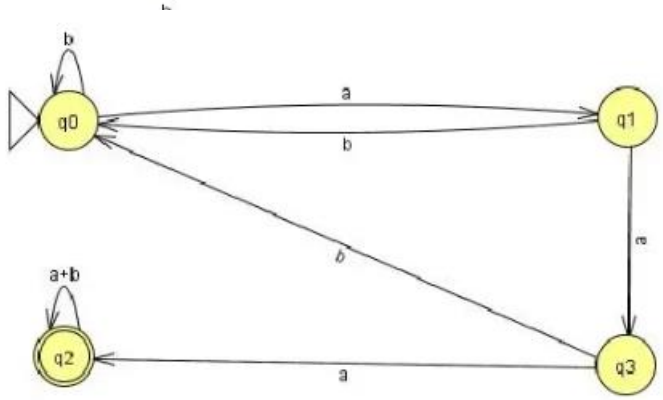

ILUSTRACIÓN 2: PASO 1

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

APLICACIÓN: Al aplicar el paso 2 a nuestro DFA, podemos ver que no hay transición de q_0 a q_2, tampoco existe un bucle en q_1, tampoco hay transición de q_3 a q_2, 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:

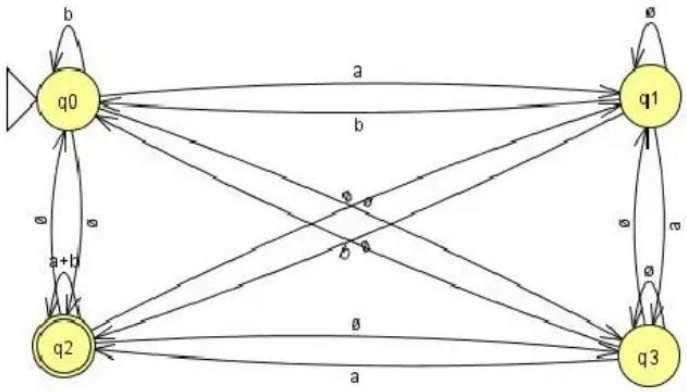

ILUSTRACIÓN 3: PASO 2

PASO 3: Seleccionar un estado q_r, talque q_r NO sea un estado inicial y/o final. Luego, por cada estado q_x se selecciona un camino, pasando por q_r, hacia cada estado q_y del DFA, talque $q_x \neq q_r$ y $q_y \neq q_r$. Ahora se crea una transición Q_j que tenga como Expresión Regular el símbolo (o Expresión Regular) de la transición que va de q_x a q_r concatenado con el símbolo de la transición que va de q_r a q_y. Al bucle que se hace en q_r se le aplicará la operación de clausura (o clausura Kleene) y se concatenará con la Expresión Regular antes encontrada. A esta nueva transición Q_j se le aplica una operación de unión con el símbolo de la transición que va de q_x a q_y. La transición Q_j deberá quedar de la forma $T_i S^* T_j + T_k$. Esta nueva transición Q_j transitará del estado q_x al estado q_y.

APLICACIÓN: Ahora bien, seleccionaremos como estado q_r a q_1. Haciendo la concatenación da cada transición desde todos los estados q_x hacia todos los estados q_y usando a q_r 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 q_r y haremos la concatenación con el Q_j 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	$b\emptyset^* \emptyset$

El siguiente paso es hacer la unión del Q_j que ya tenemos con el símbolo de la transición que va de q_x a q_y 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 q_1 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 q_0 y q_2). El DFA nos queda de la siguiente manera:

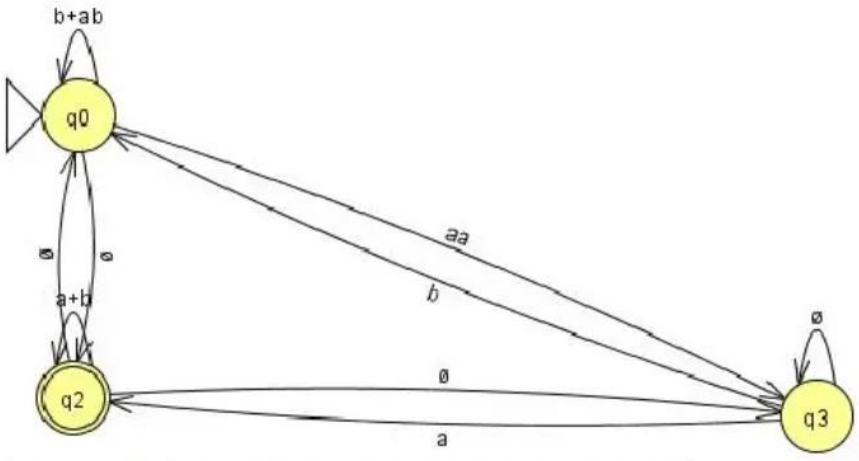

ILUSTRACIÓN 4: PASO 3

Ahora que ya hemos comprendido los pasos esenciales de la eliminación de estados, presentaremos la tabla para eliminar el estado q_3:

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:

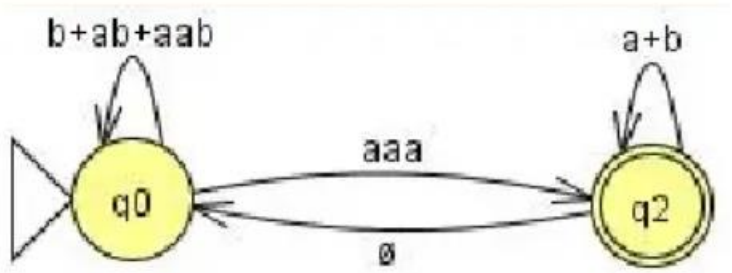

ILUSTRACIÓN 5: PASO 3

PASO 4: Si tenemos un estado inicial q_x y un estado final q_y donde $q_x \neq q_y$, se debería generar una Expresión Regular a partir de este DFA de la forma $(R + SU^*T)^*SU^*$, donde R es un bucle en q_x, S es el camino que va de q_x a q_y, U es un bucle en q_2 y T es el camino que va de q_y a q_x.

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

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

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

Algebra de las expresiones regulares

Las expresiones regulares se basan en varias reglas fundamentales que definen cómo se pueden formar y combinar. La concatenación es una de estas reglas, donde dos expresiones regulares A y B se pueden juntar para formar AB, lo que representa la

búsqueda de A seguido inmediatamente por B. Otra regla es la alternancia, denotada por A|B, que permite coincidir con A o B, actuando como un operador lógico "OR".

La clausura de Kleene, representada por A*, es otra regla importante en las expresiones regulares. Esta regla permite que el patrón A aparezca cero o más veces. De manera similar, la clausura positiva o el símbolo más (+) indica que A debe aparecer al menos una vez. La operación opcional, denotada por A?, permite que A aparezca cero o una vez, haciendo que A sea opcional.

Los paréntesis se utilizan para agrupar patrones en las expresiones regulares, lo que ayuda a definir el alcance y la precedencia de los operadores. Las clases de caracteres, indicadas por corchetes como [abc], permiten coincidir con cualquier carácter dentro de los corchetes. Los rangos también pueden especificarse dentro de las clases de caracteres, como en [a-z]. La negación de clases de caracteres se logra con un signo de intercalación al inicio, como en [^abc], que coincide con cualquier carácter excepto aquellos dentro de los corchetes.

Las anclas son símbolos especiales que coinciden con posiciones dentro de la cadena más que con caracteres específicos. Por ejemplo, ^ y $ son anclas que coinciden con el inicio y el final de una línea, respectivamente. Además, los caracteres especiales pueden escaparse utilizando una barra invertida , lo que permite que se interpreten literalmente en una expresión regular.

Estas reglas y operaciones son la base para construir expresiones regulares complejas y potentes, permitiendo un amplio rango de operaciones de búsqueda y manipulación de texto.

Bibliografía

Expresión regular _ AcademiaLab. https://academia-lab.com/enciclopedia/expresion-regular/.

Yumpu.com. (n.d.). *Como pasar de DFA a ER - csrg*. yumpu.com.

https://www.yumpu.com/es/document/read/50865741/como-pasar-de-dfa-a-er-csrg

FernandoEscher. (n.d.). *DFA a expresion regular*. Scribd.

https://es.scribd.com/doc/12929632/DFA-a-Expresion-Regular

IBM documentation. (2023, October 10).

https://www.ibm.com/docs/es/i/7.5?topic=expressions-regular