Universidad autónoma de Chiapas

Ingeniería y tecnología de software

COMPILADORES

Alumna Estrella Ailen Gonzalez Díaz

27/01/24

Expresiones regulares

I.- Explicar los tipos de operadores de expresiones regulares.

En cómputo teórico y teoría de lenguajes formales, una expresión regular o expresión racional (también son conocidas como regex o regexp, por su contracción de las palabras inglesas regular expression) es una secuencia de caracteres que conforma un patrón de búsqueda.

Las expresiones regulares son patrones utilizados para encontrar una determinada combinación de caracteres dentro de una cadena de texto. Las expresiones regulares proporcionan una manera muy flexible de buscar o reconocer cadenas de texto.

Por ejemplo, la expresión regular /[a-z]/ coincide con cualquier letra minúscula del alfabeto inglés. La expresión regular /[0-9]/ coincide con cualquier dígito numérico. La expresión regular /[a-z][0-9]/ coincide con cualquier combinación de una letra minúscula y un dígito numérico

Los operadores de expresiones regulares se utilizan para combinar caracteres y patrones para crear expresiones regulares más complejas. Los tipos de operadores de expresiones regulares más comunes son los siguientes:

Operadores de caracteres individuales: Estos operadores coinciden con un solo carácter específico. Por ejemplo, el operador a coincide con la letra a.

Operadores de rango: Estos operadores coinciden con un rango de caracteres. Por ejemplo, el operador [a-z] coincide con cualquier letra minúscula del alfabeto inglés.

Operadores de repetición: Estos operadores indican que un patrón se puede repetir un número determinado de veces. Por ejemplo, el operador a* coincide con cero o más ocurrencias de la letra a.

Operadores de agrupación: Estos operadores se utilizan para agrupar patrones para su manipulación posterior. Por ejemplo, el operador (a|b) coincide con la letra a o la letra b

II.- Explicar el proceso de conversión de DFA a expresiones regulares.

El proceso de conversión de un autómata finito determinista (AFD) a una expresión regular es un proceso sencillo que se puede realizar en los siguientes pasos:

1. Identificar el alfabeto del AFD. El alfabeto es el conjunto de símbolos que el AFD puede leer.

2. Identificar el estado inicial del AFD. El estado inicial es el estado desde el cual el AFD comienza a leer una cadena.

3. Identificar los estados de aceptación del AFD. Los estados de aceptación son los estados en los que el AFD termina cuando se lee una cadena que pertenece al lenguaje que reconoce.

4. Asignar un símbolo a cada transición. A cada transición del AFD se le asigna un símbolo del alfabeto.

Una vez que se han completado estos pasos, se puede comenzar a construir la expresión regular. La expresión regular se construye de la siguiente manera:

Se comienza con el símbolo correspondiente al estado inicial.

Se repite el siguiente paso hasta que se alcance un estado de aceptación:

Si la transición desde el estado actual es a un estado de aceptación, se agrega el símbolo correspondiente a la transición a la expresión regular.

Si la transición desde el estado actual no es a un estado de aceptación, se agrega el símbolo correspondiente a la transición a la expresión regular, seguido de un símbolo de concatenación.

Por ejemplo, considere el siguiente AFD:

q0

/ \

q1 q2

El alfabeto del AFD es {0, 1}. El estado inicial es q0. Los estados de aceptación son q1 y q2. Las transiciones del AFD son las siguientes:

* q0 → q1, 0

* q0 → q2, 1

* q1 → q1, 0

* q1 → q2, 1

* q2 → q2, 0

* q2 → q1, 1

La expresión regular que corresponde a este AFD es la siguiente:

(0 + 1)(0 + 1)*

La expresión regular se construye de la siguiente manera:

Comienza con el símbolo correspondiente al estado inicial, q0: (0 + 1)

Repite el siguiente paso hasta que se alcance un estado de aceptación:

La transición desde el estado q0 es a los estados q1 y q2. Ambos estados son estados de aceptación, por lo que se agrega el símbolo correspondiente a la transición a la expresión regular: (0 + 1)

La transición desde el estado q1 es a sí mismo o al estado q2. La transición a sí mismo no es a un estado de aceptación, por lo que se agrega el símbolo correspondiente a la transición a la expresión regular, seguido de un símbolo de concatenación: (0 + 1)(0 + 1)

La transición desde el estado q2 es a sí mismo o al estado q1. La transición a sí mismo no es a un estado de aceptación, por lo que se agrega el símbolo correspondiente a la transición a la expresión regular, seguido de un símbolo de concatenación: (0 + 1)(0 + 1)*

La expresión regular (0 + 1)(0 + 1)* reconoce el lenguaje de todas las cadenas de 0 y 1 que comienzan con 0 o 1, y que pueden contener cualquier número de 0 y 1.

Cabe señalar que el proceso de conversión de AFD a expresiones regulares también se puede realizar utilizando un algoritmo. El algoritmo más común es el algoritmo de Thompson.

III.- Explicar leyes algebraicas de expresiones regulares.

Las leyes algebraicas de expresiones regulares son un conjunto de reglas que permiten simplificar expresiones regulares sin cambiar su significado. Estas leyes se basan en las propiedades de los lenguajes regulares, que son los lenguajes que pueden ser descritos por expresiones regulares.

Las leyes algebraicas de expresiones regulares se dividen en tres categorías:

Leyes de identidad: Estas leyes establecen que dos expresiones regulares son equivalentes.

Leyes de conmutatividad: Estas leyes establecen que el orden de los operados en una expresión regular no cambia su significado.

Leyes de asociatividad: Estas leyes establecen que el orden de las operaciones en una expresión regular puede simplificarse sin cambiar su significado.

Las siguientes son algunas de las leyes algebraicas de expresiones regulares más comunes:

Ley de identidad para la unión: $r + \emptyset = r$

Ley de identidad para la concatenación: $r . \varepsilon = r$

Ley de conmutatividad para la unión: $r + s = s + r$

Ley de conmutatividad para la concatenación: $r . s = s . r$

Ley de asociatividad para la unión: $(r + s) + t = r + (s + t)$

Ley de asociatividad para la concatenación: (r . s) . t = r . (s . t)

Por ejemplo, la ley de conmutatividad para la unión establece que r + s es equivalente a s + r. Esto significa que las expresiones regulares a + b y b + a describen el mismo lenguaje, que es el conjunto de todas las cadenas que contienen al menos un símbolo a o un símbolo b.

Las leyes algebraicas de expresiones regulares se pueden utilizar para simplificar expresiones regulares complejas. Por ejemplo, la expresión regular (a + b) * (c + d) puede simplificarse utilizando la ley de conmutativa para la unión y la ley asociativa para la concatenación como sigue:

(a + b) * (c + d)

= (a * (c + d)) + (b * (c + d))

= (ac + ad) + (bc + bd)

La expresión regular simplificada, ac + ad + bc + bd, describe el mismo lenguaje que la expresión original.

Las leyes algebraicas de expresiones regulares son una herramienta importante para trabajar con expresiones regulares. Se utilizan en una variedad de aplicaciones, como la programación de computadoras, el análisis de texto y la teoría de la computación.