TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Pachuca

Materia: Lenguajes y Autómatas I

Docente: Rodolfo Baumé Lazcano

Nombre de la actividad: Investigación expresiones regulares

Tema: Expresiones regulares

Alumna: Yesenia Morales Ordoñez

Fecha de entrega: 19-Marzo-2024

Introducción

Las expresiones regulares son un poderoso y versátil concepto utilizado en la teoría de la computación y la programación para describir patrones de cadenas de texto. Se utilizan en una variedad de aplicaciones, desde la búsqueda y manipulación de texto hasta la validación de datos y la construcción de compiladores. En el contexto de los lenguajes formales y los autómatas, las expresiones regulares sirven como una herramienta fundamental para definir conjuntos de cadenas que cumplen ciertos criterios o patrones específicos.

Un autómata es un modelo matemático abstracto que representa un sistema que puede cambiar de un estado a otro en respuesta a una secuencia de entradas de acuerdo con un conjunto de reglas definidas. Los autómatas finitos y los autómatas de pila son ejemplos comunes de autómatas utilizados en la teoría de la computación.

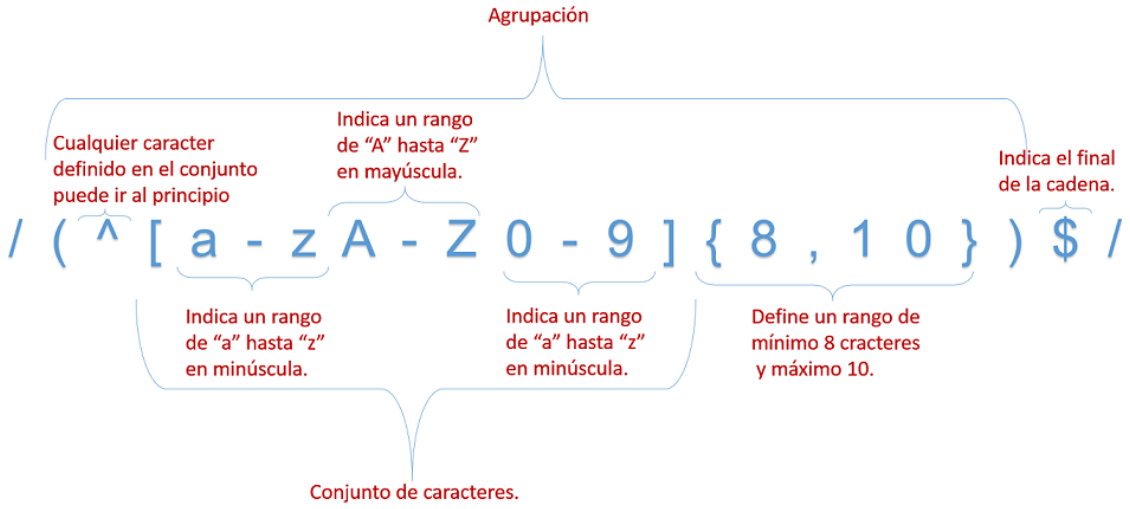
Las expresiones regulares se utilizan para describir los conjuntos de cadenas aceptadas por un autómata o para definir patrones de búsqueda en un texto. Estas expresiones son construcciones algebraicas que representan secuencias de caracteres, metacaracteres y operadores que permiten realizar coincidencias y búsquedas eficientes dentro de textos.

¿Qué son las expresiones regulares?

Definicion: Las regex son cadenas de caracteres basadas en reglas sintácticas que permiten describir secuencias de caracteres. Así, forman parte de los lenguajes regulares, los cuales son un subgrupo de los lenguajes formales, de gran importancia para la tecnología de la información y, especialmente, para el desarrollo de software.

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 esto 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.



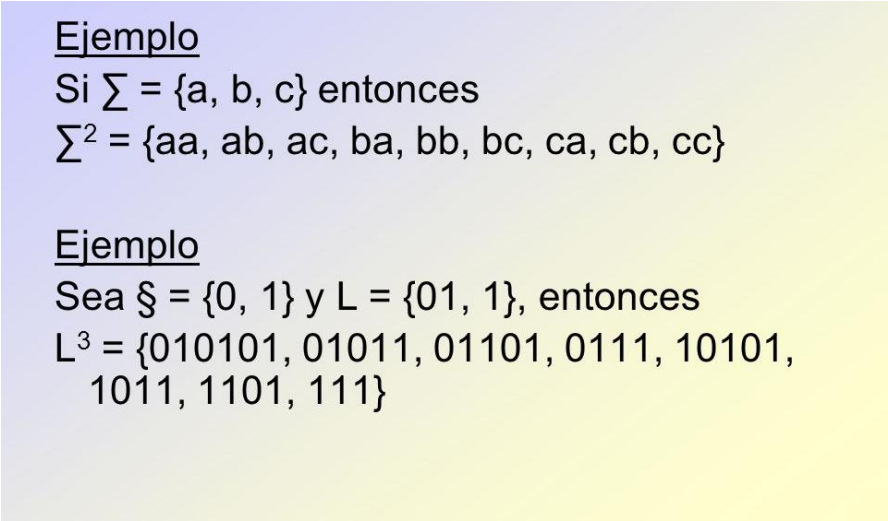
¿Cómo funciona una expresión regular?

Una expresión regular puede estar formada, o bien exclusivamente por caracteres normales (como abc), o bien por una combinación de caracteres normales y metacaracteres (como ab*c). Los metacaracteres describen ciertas construcciones o disposiciones de caracteres: por ejemplo, si un carácter debe estar en el inicio de la línea o si un carácter solo debe o puede aparecer exactamente una vez, más veces o menos. Ambos ejemplos de expresiones regulares funcionan, por ejemplo, de la siguiente manera:

abc. El patrón regex sencillo abc requiere una coincidencia exacta. Por tanto, se buscarán cadenas de caracteres que no solo contengan los caracteres “abc”, sino que también aparezcan en ese orden. Una pregunta como “¿Conoces la plaza ABC?” ofrece la coincidencia buscada por esta expresión.

ab*c. Las expresiones regulares con caracteres especiales funcionan de manera diferente, ya que no solo se buscarán coincidencias exactas, si no también escenarios especiales. En este caso, el asterisco hace que la búsqueda se centre en cadenas de caracteres que empiecen por la letra “a” y que terminen por la letra “c” y entremedias cuenten con cualquier número de caracteres “b”. Así se mostrará como coincidencia tanto “abc”, como la cadena de caracteres “abbbbc” y “cbbabbcba”.

Además, cada regex se puede vincular a una acción concreta, como la ya mencionada “Reemplazar”. Esta acción se ejecuta en todos los lugares en los que se detecta la expresión regular, es decir, en todos los puntos en los que haya una coincidencia similar a la de los ejemplos.



El punto "."

El punto se interpreta por el motor de búsqueda como "cualquier carácter", es decir, busca cualquier carácter incluyendo los saltos de línea. Los motores de expresiones regulares tienen una opción de configuración que permite modificar este comportamiento. En .Net Framework se utiliza la opción RegexOptions.Singleline para especificar la opción de que busque todos los caracteres incluidos el salto de línea (\n).

El punto se utiliza de la siguiente forma: Si se le dice al motor de RegEx que busque g.t en la cadena "el gato de piedra en la gótica puerta de getisboro goot" el motor de búsqueda encontrará "gat", "gót" y por último "get". Nótese que el motor de búsqueda no encuentra "goot"; esto es porque el punto representa un solo carácter y únicamente uno. Si es necesario que el motor encuentre también la expresión "goot", será necesario utilizar repeticiones, las cuales se explican más adelante.

El signo de exclamación "!"

Se utiliza para realizar una "búsqueda anticipada negativa". La construcción de la expresión regular es con el par de paréntesis, el paréntesis de apertura seguido de un signo de interrogación y un

signo de exclamación. Dentro de la búsqueda tenemos la expresión regular. Por ejemplo, para excluir exactamente una palabra, habrá que utilizar ^(palabra.+|(?!palabra).*)$.

Los corchetes "[ ]"

La función de los corchetes en el lenguaje de las expresiones regulares es representar "clases de caracteres", o sea, agrupar caracteres en grupos o clases. Son útiles cuando es necesario buscar uno de un grupo de caracteres. Dentro de los corchetes es posible utilizar el guion - para especificar rangos de caracteres. Adicionalmente, los metacaracteres pierden su significado y se convierten en literales cuando se encuentran dentro de los corchetes. Por ejemplo, como vimos en la entrega anterior \d es útil para buscar cualquier carácter que represente un dígito. Sin embargo esta denominación no incluye el punto . que divide la parte decimal de un número. Para buscar cualquier carácter que representa un dígito o un punto podemos utilizar la expresión regular [\d.].

La barra "|"

Sirve para indicar una de varias opciones. Por ejemplo, la expresión regular a|e encontrará cualquier "a" o "e" dentro del texto. La expresión regular este|oeste|norte|sur permitirá encontrar cualquiera de los nombres de los puntos cardinales. La barra se utiliza comúnmente en conjunto con otros caracteres especiales.

El signo de dólar "$"

Representa el final de la cadena de caracteres o el final de la línea, si se utiliza el modo multilínea. No representa un carácter en especial sino una posición. Si se utiliza la expresión regular \.$ el motor encontrará todos los lugares donde un punto finalice la línea, lo que es útil para avanzar entre párrafos.

El acento circunflejo "^"

Este carácter tiene una doble funcionalidad, que difiere cuando se utiliza individualmente y cuando se utiliza en conjunto con otros caracteres especiales. En primer lugar su funcionalidad como carácter individual: el carácter ^ representa el inicio de la cadena (de la misma forma que el signo de dólar $ representa el final de la cadena). Por tanto, si se utiliza la expresión regular ^[a-z] el motor encontrará todos los párrafos que den inicio con una letra minúscula. Cuando se utiliza en conjunto con los corchetes de la siguiente forma [^\w] permite encontrar cualquier carácter que NO se encuentre dentro del grupo indicado. La expresión indicada permite encontrar, por ejemplo, cualquier carácter que no sea alfanumérico o un espacio, es decir, busca todos los símbolos de puntuación y demás caracteres especiales.

Los paréntesis "()"

De forma similar que los corchetes, los paréntesis sirven para agrupar caracteres, sin embargo existen varias diferencias fundamentales entre los grupos establecidos por medio de corchetes y los grupos establecidos por paréntesis:

- Los caracteres especiales conservan su significado dentro de los paréntesis.

- Los grupos establecidos con paréntesis establecen una "etiqueta" o "punto de referencia" para el motor de búsqueda que puede ser utilizada posteriormente como se denota más adelante.
- Utilizados en conjunto con la barra | permite hacer búsquedas opcionales. Por ejemplo la expresión regular al (este|oeste|norte|sur) de permite buscar textos que den indicaciones por medio de puntos cardinales, mientras que la expresión regular este|oeste|norte|sur encontraría "este" en la palabra "esteban", no pudiendo cumplir con este propósito.
- Utilizados en conjunto con otros caracteres especiales que se detallan posteriormente, ofrece funcionalidad adicional.

El signo de interrogación "?"

El signo de interrogación tiene varias funciones dentro del lenguaje de las expresiones regulares. La primera de ellas es especificar que una parte de la búsqueda es opcional. Por ejemplo, la expresión regular ob?scuridad permite encontrar tanto "oscuridad" como "obscuridad". En conjunto con los paréntesis redondos permite especificar que un conjunto mayor de caracteres es opcional; por ejemplo Nov(\.|iembre|ember)? permite encontrar tanto "Nov" como "Nov.", "Noviembre" y "November".

Las llaves "{}"

Comúnmente las llaves son caracteres literales cuando se utilizan por separado en una expresión regular. Para que adquieran su función de metacaracteres es necesario que encierren uno o varios números separados por coma y que estén colocados a la derecha de otra expresión regular de la siguiente forma: \d{2} Esta expresión le dice al motor de búsqueda que encuentre dos dígitos contiguos. Utilizando esta fórmula podríamos convertir el ejemplo ^\d\d/\d\d/\d\d\d\d$ que servía para validar un formato de fecha en ^\d{2}/\d{2}/\d{4}$ para una mayor claridad en la lectura de la expresión.

El asterisco "*"

El asterisco sirve para encontrar algo que se encuentra repetido 0 o más veces. Por ejemplo, utilizando la expresión [a-zA-Z]\d* será posible encontrar tanto "H" como "H1", "H01", "H100" y "H1000", es decir, una letra seguida de un número indefinido de dígitos. Es necesario tener cuidado con el comportamiento del asterisco, ya que este, por defecto, trata de encontrar la mayor cantidad posible de caracteres que correspondan con el patrón que se busca. De esta forma si se utiliza \(.*\) para encontrar cualquier cadena que se encuentre entre paréntesis y se lo aplica sobre el texto "Ver (Fig. 1) y (Fig. 2)" se esperaría que el motor de búsqueda encuentre los textos "(Fig. 1)" y "(Fig. 2)", sin embargo, debido a esta característica, en su lugar encontrará el texto "(Fig. 1) y (Fig. 2)". Esto sucede porque el asterisco le dice al motor de búsqueda que llene todos los espacios posibles entre los dos paréntesis. Para obtener el resultado deseado se debe utilizar el asterisco en conjunto con el signo de interrogación de la siguiente forma: \(.*?\) Esto es equivalente a decirle al motor de búsqueda que "Encuentre un paréntesis de apertura y luego encuentre cualquier secuencia de caracteres hasta que encuentre un paréntesis de cierre".

El signo de suma "+"

Se utiliza para encontrar una cadena que se encuentre repetida una o más veces. A diferencia del asterisco, la expresión [a-zA-Z]\d+ encontrará "H1" pero no encontrará "H". También es posible utilizar este metacarácter en conjunto con el signo de interrogación para limitar hasta donde se efectúa la repetición.

Importancia de las expresiones regulares

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.

Casos de uso de las expresiones regulares

Coincidencia con cualquier dirección de correo de un determinado dominio	
Ejemplo de uso	Coincidencia con cualquier dirección de correo electrónico de los dominios **yahoo.com**, **hotmail.com** y **gmail.com**.
Ejemplo de regex	(\W|^)[\w.\-]{0,25}@(yahoo|hotmail|gmail)\.com(\W|$)

Coincidencia con un formato alfanumérico	
Ejemplo de uso	Coincidencia con los números de pedido de la empresa. Hay varios formatos posibles; por ejemplo: • PO nn-nnnnn • PO-nn-nnnn • PO# nn nnnn • PO#nn-nnnn • PO nnnnnn
Ejemplo de regex	(\W|^)po[#\-]{0,1}\s{0,1}\d{2}[\s-]{0,1}\d{4}(\W|$)

Coincidencia con cualquier dirección IP de un intervalo de direcciones	
Ejemplo de uso	Coincidencia con cualquier dirección IP que se incluya en el intervalo de **192.168.1.0** a **192.168.1.255**.
Ejemplos de expresiones regex	**Ejemplo 1:** 192\.168\.1\. **Ejemplo 2:** 192\.168\.1\.\d{1,3}

Conclusiones

En conclusión, las expresiones regulares son una herramienta poderosa y versátil en la teoría de la computación y la programación. Permiten describir patrones de cadenas de texto de manera concisa y eficiente, lo que facilita la búsqueda, manipulación y validación de datos en una amplia gama de aplicaciones. En el contexto de los lenguajes formales y los autómatas, las expresiones regulares son fundamentales para definir conjuntos de cadenas aceptadas por un autómata y para diseñar sistemas que puedan reconocer y procesar textos de manera automática. Su uso es fundamental tanto en la teoría como en la práctica de la informática, y comprender su funcionamiento y aplicaciones es esencial para cualquier persona que trabaje en el campo de la programación y la ciencia de la computación.

Referencias

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

colaboradores de Wikipedia. (2024b, febrero 1). *Expresión regular*. Wikipedia, la Enciclopedia Libre. https://es.wikipedia.org/wiki/Expresi%C3%B3n_regular#:~:text=Las%20expresiones%20regulares%20son%20patrones,o%20reconocer%20cadenas%20de%20texto.

Ejemplos de expresiones regulares - Ayuda de Administrador de Google Workspace. (s. f.). https://support.google.com/a/answer/1371417?hl=es