

MEMORIA PRACTICA MODELOS DE COMPUTACION

CEU SanPablo Ingenieria Informatica

Juan Federico García
Alonso-Burón

INDICE:

1-Introduccion a la practica

2-JFlex

- JFlex

- Grafo

- Funcionamiento del autómata

3-JSoup

- JSoup

4-Test

5-Conclusiones

1- Introducción a la practica

En esta practica se ha desarrollado un programa basado en un autómata finito determinista, cuya función sea analizar una pagina web reconociendo la estructura de esta, y obtener una serie de links basándose en una serie de filtros. Además, también permite reconocer si una página esta balanceada.

Durante el desarrollo de la practica, se han empleado dos enfoques en la programación de este. Estos enfoques son: JFlex, y JSoup.

Para desarrollar esta practica, hemos empleado Net Beans, un entorno de desarrollo compatible con todos los recursos que se iban a utilizar.

2- JFlex

-JFlex

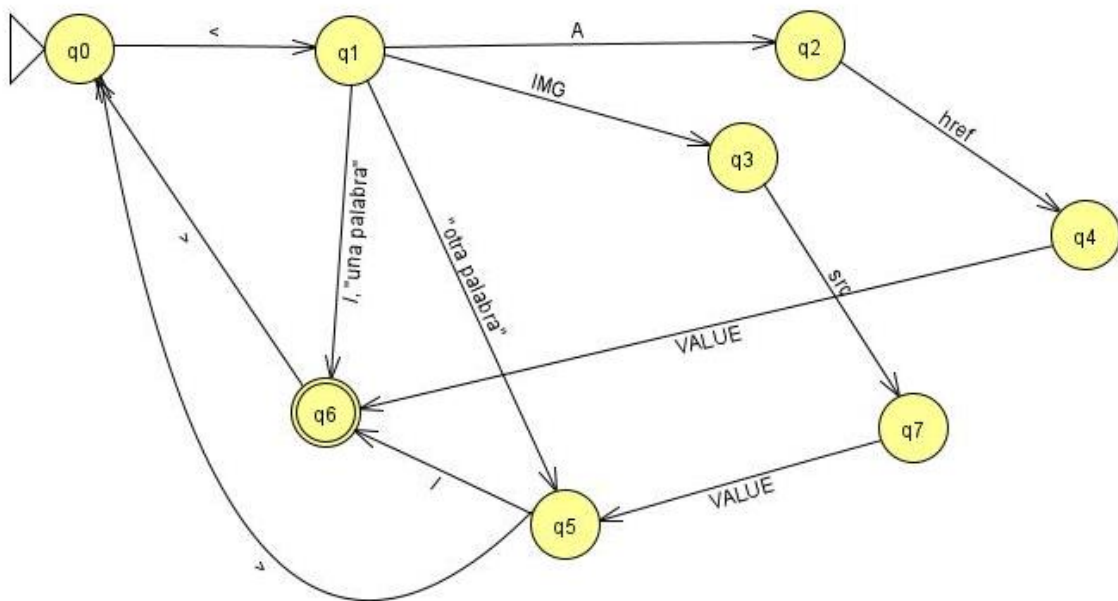
Durante la practica hemos estado utilizando JFlex, una herramienta que permite la generación de un analizador léxico en java.

Esta herramienta ha sido especialmente útil para reconocer que tipo de caracteres leía el ordenados, lo que nos permite clasificarlos por tokens, y así poder crear el autómata.



-Grafo

En la foto que se muestra a continuación, se puede observar el grafo del autómata que he diseñado y utilizado, donde se pueden apreciar todos los estados y trnasiciones que este pose.



-Funcionamiento del autómata

ESTADOS:

Q0: Si reconoce un token OPEN ("`<`"), el autómata se inicializa, y transita al estado q1.

Q1: Una vez se ha inicializado el autómata, empieza a reconocer token. En caso de que reciba un token de tipo WORD, lo analiza y obtiene el valor de este. Si reconoce "A", añade a la pila el token, y transita a q2, si reconoce "IMG", añade el token a la pila y transita a Q3, y si no coincide con ninguno de estos dos casos, añade el token a la pila y transita a q5. En el caso de que entre un token de tipo SLASH, lee el siguiente token, y si valor del token coincide con el valor del token de la pila, elimina el token de la pila y transita a q6. Si se da otro caso, el autómata se pararía y descartaría la sentencia.

Q2: Si recibe el token tipo WORD, y el valor de este es "href", lee el siguiente token, y si es de tipo EQUAL, transita a q4.

Q3: Si el autómata recibe un token de tipo WORD, y el valor de este es "src", transita a q7.

Q4: El autómata recibe un token de tipo VALUE, y lo almacena en una lista denominada "urlsA", y transita a q6.

Q5: Al recibir un token de tipo CLOSE, se reiniciaría el autómata (Esto daría como valida la sentencia).

Q6: En caso de recibir un token de tipo CLOSE, transita a q0 reiniciando el autómata.

Q7: Si llega un token de tipo VALUE, añade el valor del token a la lista "urlsIMG", y transita a q5.

3- JSoup

JSoup

Jsoup es una librería que nos permite construir a partir de una URL de un documento HTML un árbol DOM que refleje su contenido. El árbol que obtenemos nos permite filtrar y, por lo tanto obtener los elementos que deseemos de la URL.

Para obtener los elementos que deseábamos (links de tipo A, y links de imágenes), hemos empleado una estructura similar en ambos métodos.

La consistía en: inicializar un ArrayList denominado "resultado", después creamos una lista de elementos con el método `webPage.select("A")`, y posteriormente lo volcábamos en "resultado" haciendo uso de un bucle "for" filtrando los elementos con el atributo "href", y posteriormente, devolvíamos "resultado" con un "return".


Para obtener los links de tipo IMG, la estructura es similar, solo que el método `webPage.select("A")`, en vez de filtrar con "A", filtraba con "IMG", y pasa algo similar con el atributo dentro del bucle for, que pasa de ser "href", a "src".


4- Test


Durante todo el proyecto, el desarrollo ha estado sometido a varios test, que indicaban si el funcionamiento del código era correcto. Existían varios test variados. El primer tipo de test iba dirigido a JFlex, que comprobaba que devolviese los links deseados, y que demostrase si el balanceo que se da es el correcto. El segundo tipo de test, Se centraba en JSoup, que comprobaba que obtuviésemos todos los links que deseábamos. El ultimo tipo de test, eran unos test cruzados, que se aseguraban que se pasaban todos los test de JSoup y de JFlex.


Tests passed: 100,00 %


All 15 tests passed. (0,843 s)





 es.ceu.gisi.modcomp.webcrawler.tests.crosstests.CrossTest **passed**




 es.ceu.gisi.modcomp.webcrawler.tests.htmlparser.HTMLParserTest **passed**



 es.ceu.gisi.modcomp.webcrawler.tests.jflex.JFlexScraperTest **passed**



 es.ceu.gisi.modcomp.webcrawler.tests.jsoup.JsoupScraperTest **passed**

5- Conclusiones

Este proyecto ha ayudado a entender el funcionamiento de los autómatas, y sobre todo como poder plasmarlo en un programa. Durante las aproximadas 14 horas que he empleado en desarrollar la practica, las 5 primeras fueron conjuntas con mi compañero Tomas Machin.