Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computación

Documentación: Analizador semántico

Curso: Compiladores e Intérpretes
Erick Castillo Quesada
201138074
Leonardo Mendoza Fernández
201101376

Descripción del problema

El problema general para el proyecto consiste en tomar el árbol generado anteriormente y revisar que este este bien estructurado, y que cada uno de sus elementos estén bien ubicados según el contexto en el tree. Para esto se deberán hacer ciertos cambios en el scanner y el parcer ya definidos, para que de esta forma se puedan realizar un analizador óptimo.

Este problema general se puede dividir en algunos problemas especificaos como:

1-Definir todos los elementos que deben ser revisados
2-Definir el contexto bajo el cual cada uno de los elementos deben encontrarse
3-Definir elementos obsoletos para html5
3-Comprender la estructura semántica de XHTML
4-Imprimir los errores con su ubicación posicional en el XHTML original
5-Crear algoritmos tanto de revisión como de recorrido de nodos

Una ves comprendido el problema y los problemas específicos que este implica se inicia con la realización de la solución. Que requiere la investigación de la valides contextual de los elementos y encontrar métodos ideales para recorrer y revisar el árbol.

Diseño del programa

Adaptación del scanner

El scanner ya se encontraba optimizado bastante bien para el fin de este programa, pero ahora se le agrego la capacidad de identificar urls. De esta forma es muy fácil poder identificar los nodos de tipo url en el árbol y validar su correcta ubicación. Por lo tanto no es necesario revisar todos los argumentos de tipo texto cuando se espera un url.

Adaptación del parser

Para imprimir la posición en caso de error de un forma más sencilla cada nodo del árbol ahora contiene la posición filas columnas en la que se encontraría el punto que representa en el documento original. Además se redujo la cantidad de nodos hijos producidos, y de esta forma es más fácil hacer el recorrido.

Creación de analizador del árbol

Definición del algoritmo

Una vez que se a finalizado la construcción del árbol es llamada la función que se encarga de imprimirlo. Y luego se utilizan los elementos de este modulo para analizarlo semánticamente. Semejante a como se ilustra en el siguiente algoritmo.

```
AnálisisDeArbol (Arbol){

    For ( Nodo in Arbol){
          if (Nodo contiene etiqueta revisable){
                for (Atributo in Nodo){
                     RevisaValidesDeAtributo();
                      if (Existen errores) imprima Error;
                     for (Argumentos in Atributo){
                              RevisaValidesdeArgumento();
                               if (Existen errores) imprima Error;
                     }
                }
          }
          AnálisisDeArbol(Nodo)
    }
}
```

Por lo tanto todo el arbol es recorrido recursivamente y se revisan todos los casos necesarios para probar que el archivo de tipo xhtml tenga valides semántica.

Definición de elementos revisados y su contexto

Para definir que etiquetas se revisan y cuales argumentos son esperados para cada una se utiliza un matriz de todas las posibles etiquetas con sus posibles atributos que es utilizada por los algoritmos de revisión de atributos para verificar la valides de esto según el contexto,

esta matriz se distribuye de la siguiente forma:

Etiqueta1	PosibleAtributo1	PosibleAtributo2	PosibleAtributoN
Etiqueta2	PosibleAtributo1	PosibleAtributo2	PosibleAtributoN
....	....		
EtiquetaN	PosibleAtributo1	PosibleAtributo2	PosibleAtributoN

Por lo tanto la primera columna contiene todas las etiquetas que son revisadas, y el las siguientes columnas contienen los atributos permitidos

se usa un método similar para definir los argumentos esperado para cada atributo

Atributo1	PosibleArgumento1	PosibleArgumento2	...	PosibleArgumentoN
Atributo2	PosibleArgumento1	PosibleArgumento2	...	PosibleArgumentoN
....	....			
AtributoN	PosibleArgumento1	PosibleArgumento2	...	PosibleArgumentoN

Por lo tanto la primera columna contiene todos los posibles atributos, y las siguientes columnas contiene los argumentos validos para cada uno.

En esta matriz existen casos diferentes
1 Casos estáticos:
Estos casos simplemente se busca que el argumento sea alguno de los textos específicos como el caso de formmethod se requiere que el texto sea o get o post

2 Casos de formato
En estos casos se busca cierto formato para el argumento, actualmente se soportan fechas, números, textos, lenguajes y urls, estos casos se representan con un reglón al inicio de casa posible argumento.

Analisis de resultados

Se considera que la realización de proyecto fue un éxito ya que se logra alcanzar todos los objetivos. Pues se logro definir una forma óptima de lo que se esperaba en el árbol. Y además se logro hacer el recorrido y revisarlo según lo definido.

Dado que se logró generar el analizador, se logró comprender a profundidad la estructura del XHTML, en especial su estructura. Debido a esto se logró definir un analizador que recursivo para revisar si un árbol es semánticamente válido o no.

En cuanto a tiempo, se realizó una distribución adecuada de este dado que no se presentaron problemas en cuanto a la finalización de la tarea después de la fecha límite.

Manual de usuario

Una vez abierta la terminal, y al cambiar la dirección a la de la carpeta Parser_XHTML, sólo se deben de correr dos instrucciones para ejecutar el scanner.
El primero permite generar el archivo parser y consiste sólo de la instrucción:

```
make
```

Para correr el parser se escribe la siguiente instrucción, donde nombreArchivo corresponde al archivo xhtml al que se le desea realizar un análisis sintáctico.

```
./parser nombreArchivo
```

El parser imprimirá el parse tree relacionado al archivo o los errores si se presenta el caso.

Conclusión personal

En sí, la tarea programada permitió reforzar los términos vistos en clase sobre análisis semántico, dando un refuerzo al contenido visto previamente. Además, permitió una comprensión más clara sobre el funcionamiento de analizador sintáctico, semántico y léxico y los algoritmos asociados.

A pesar de no ser considerada, una tarea programada tan pesada y que no representó una dificultad como otras previamente realizadas en cursos anteriores, debido a su naturaleza, en especial la necesidad de definir el contexto semántico, se requirió de gran investigación y dedicación, para crear sin errores las producciones que revisaría una estructura válida para un documento XHTML.

En cuanto al lenguaje, debido a que C posee un manejo de strings diferentes ya que estos consisten en arreglos de char y se debe manejar la memoria y alocar de manera correcta estos y hacer copias para evitar que se apunte a un arreglo de caracteres variable. Esto presentó grandes complicaciones ya que generaba segmentation fault y requirió tiempo para aprender a manejar todas estas alocaciones ya que no se tenía suficiente práctica con C. Sin embargo, fue un importante obstáculo que permitió que se conociera mejor C para futuros usos.

En conclusión, se considera que la tarea, posee todas las cualidades para ir de la mano con lo visto en clase y sirve como base para conocer un un caso específico de analizador, pero a su vez brinda una comprensión general de este.

Referencias

Axelsson,J et all. XHTML 2.0. Recuperado de http://www.w3.org/TR/xhtml2/Overview.html#toc

Carreras, O. (2007) Plantilla base XHTML. Recuperado de http://olgacarreras.blogspot.com/2007/02/plantilla-base-xhtml.html

Donnelly, C; Stallman, R. Bison Recuperado de http://es.tldp.org/Manuales-LuCAS/BISON/bison-es-1.27.html

Eguiluz, J. Introducci[on a XHTML (Tercera Edición). Recuperado de http://librosweb.es/xhtml/

Fuentes, A. (2013) Tarea Programada 2.
Hammond, D. (2013). Beware of XHTML. Recuperado de http://www.webdevout.net/articles/beware-of-xhtml

Kyrnin, J. Web Design / HTML. Recuperado de http://webdesign.about.com/od/htmltags/p/bltags_dd.htm

Levine, J. (2009) Error Reportin and Recovery: Chapter 8 – flex & bison. Recuperado de http://shop.oreilly.com/product/9780596155988.do

Sintes,M. (2010). Lista de etiquetas XHTML 1.0. Recuperado de http://www.mclibre.org/consultar/amaya/xhtml/xhtml_etiquetas.html

JAVASCRIPTKIT. XHTML. Examples Recuperado de http://www.javascriptkit.com/howto/xhtml_intro4.shtml

W3Schools. HTML-XHTML. Recuperado de http://www.w3schools.com/html/html_xhtml.asp

XHTML.com (2008) Extensible HyperText Markup Language. Recuperado de http://xhtml.com/en/xhtml/reference