TRABAJO PRÁCTICO GRUPAL

Objetivo general

Aplicar los conceptos de expresiones regulares, gramáticas libres de contexto y gramáticas con atributos en un caso práctico mediante la utilización de la herramienta ANTLR.

Introducción

Un **Autómata finito** es un sistema que realiza cómputos sobre una entrada para producir una salida. Está formado por una 5-tupla: un alfabeto, un conjunto de estados finito, una función de transición, un estado inicial y un conjunto de estados finales:

- Q: conjunto finito de estados
- Σ: alfabeto finito
- q0 ϵ Q: estado inicial
- f: función de transición
- F ⊆ Q: conjunto de estados finales

Su funcionamiento consiste en que el autómata recibe una cadena de caracteres que pertenecen al alfabeto y va leyendo dicha entrada para procesarla según la función de transición. En caso de que la cadena pertenezca al lenguaje reconocido por el autómata, al procesar toda la cadena de caracteres el autómata alcanza un estado final.
El objetivo del Trabajo Práctico es construir un procesador (lexer + parser) de especificaciones textuales de autómatas finitos. Informalmente, el lenguaje deberá permitir especificar el tipo de autómata finito (Determinista – AFD, No Determinista – AFN, o No Determinista con transiciones épsilon – AFN-ε), la tupla que define al autómata y sus componentes. En la Figura 1 se incluye un esquema de la definición de un autómata finito según el lenguaje propuesto.

TipoDeAutómata NombreDelAutómata = (ConjuntoDeEstados, Alfabeto, FunciónDeTransición, estadoInicial, EstadosDeAceptación)
Metadatos
Definición de ConjuntoDeEstados
Definición de Alfabeto
Definición de EstadosDeAceptación
Definición de la FunciónDeTransición

Figura 1

TipoDeAutómata es la palabra *AFD*, *AFN*, o *AFN-\epsilon*.
NombreDelAutómata es una palabra de longitud mayor o igual a 1, su primer símbolo es una *letra mayúscula*, luego, es seguida de *letras*, *dígitos* y/o *guiones bajos*.
ConjuntoDeEstados identifica al conjunto de estados, es una palabra de longitud mayor o igual a 1, cuyo primer símbolo es una *letra mayúscula*, y luego, es seguido de *letras*, *dígitos* y/o *guiones bajos*.
Alfabeto identifica al alfabeto, conjunto de símbolos de entrada, es la palabra *\Sigma*.
FunciónDeTransición especifica el nombre de la función de transición del autómata, puede ser la palabra *f* o la palabra *\delta*.
estadoInicial, es una palabra que especifica al estado inicial del autómata.
EstadosDeAceptación identifica al conjunto de estados de aceptación; es una palabra de longitud mayor o igual a 1, su primer símbolo es una *letra mayúscula*, luego, y es seguido de *letras*, *dígitos* y/o *guiones bajos*.

Metadatos, permite definir el autor del autómata, fecha de su creación y versión. Está compuesto de las siguientes 3 líneas:
@autor: NombreDeAutor
@fecha: dd/mm/aaaa
@version: vMayor.vMenor.revision-opcional
NombreDeAutor es una secuencia de caracteres alfabéticos entre *comillas simples*, pueden contener *espacios en blanco* para separar los nombres y el apellido del autor.
dd/mm/aaaa es la fecha de creación, donde *dd*, *mm* y *aaaa* son *dígitos* entre el *0* y el *9*.

vMayor, *vMenor* y *revisión* son *números naturales*, *opcional* es opcional y, si es incorporado, es la palabra *alpha*, *beta*, o *rc*. Estas últimas representan: *alpha* - fase inicial de desarrollo, muy inestable, *beta* - el desarrollo avanza, estas versiones ya son aptas para realizar las pruebas, *rc* - release candidate, versión que salvo cambios de última hora acabará siendo la versión final.

Definición de ConjuntoDeEstados permite definir el conjunto de estados por extensión, por lo que debe incluir el nombre de *ConjuntoDeEstados* el símbolo igual (=) y luego enumerar los *estados* entre *llaves*, al menos un estado debe existir.
ConjuntoDeEstados = {estado1, estado2, estado3, estadon}
Cada $estado_i$ es identificado por una palabra de longitud mayor o igual a 2, inicia con una "*q*", y luego es seguida de *letras*, *dígitos* y/o *guiones bajos*.

Definición de Alfabeto permite definir el conjunto de símbolos por extensión, por lo que debemos incluir el nombre del Alfabeto (*Sigma*) el símbolo igual (=) y luego enumerar los *símbolos* entre *llaves*, al menos un símbolo debe existir.
\Sigma = {simbolo1, simbolo2, simbolon}
Cada $símbolo_i$ es identificado por un *símbolo imprimible ASCII* (desde el 33 "*!*" al 38 "*&*", desde el 48 "*0*" al 57 "*9*", el 64 "@", y desde el 97 "*a*" al 122 "*z*").

Definición de EstadosDeAceptación permite definir el conjunto de estados de aceptación por extensión, del mismo modelo que se define *ConjuntoDeEstados*. Entonces, se debe incluir el nombre de *EstadosDeAceptación* el símbolo igual (=) y luego enumerar los *estados* entre *llaves*.
EstadosDeAceptación = {estado1, estado2, estado3, estadon}
Cada $estado_i$ es identificado por una palabra de longitud mayor o igual a 2, inicia con una "*q*", y luego es seguida de *letras*, *dígitos* y/o *guiones bajos*.

Definición de la FunciónDeTransición permite definir de manera analítica la función de transición.
FunciónDeTransición(argumentosSeparadosPorComa)= resultado
FunciónDeTransición es la palabra *f* o *\delta*.
Los *argumentosSeparadosPorComa* y *resultado* dependen del tipo de autómata.
Si el autómata es un AFD,
argumentosSeparadosPorComa de AFD es
estado,símbolo
resultado de AFD es *estado*
Si el autómata es un AFN,
argumentosSeparadosPorComa de AFN es
estado,símbolo
resultado de AFN es un conjunto por extensión de estados
{estado1, estado2, estadon}
es posible emplear *\emptyset* para definir el conjunto de estados vacío.
Si el autómata es un AFN-ε,
argumentosSeparadosPorComa de AFN-ε es
estado,símbolo o *\epsilon*
resultado de AFN-ε es un conjunto por extensión de estados
{estado1, estado2, estadon}
es posible emplear *\emptyset* para definir el conjunto de estados vacío.

Cabe destacar que se pueden emplear comentarios simples, con *//* lo que está a la derecha, hasta el *final de la línea* es un comentario.

Las Figuras 2, 3, y 4 ilustran ejemplos de definiciones de AFD, AFN, y AFN-ε, respectivamente. Los ejemplos de las figuras están disponibles en los archivos **pruebaAFD.af**, **pruebaAFN.af**, y **pruebaAFNepsilon.af**

```
AFD Acepta0Pares = (Q, \Sigma, f, q_par0, F)
@autor: 'Silvio'
@fecha: 11/04/2023
@version: 1.0.0-beta
// Definición del conjunto de estados Q
Q= {q_par0, q_impar0}
\Sigma= {0, 1}                  // definición del alfabeto
F= {q_par0}                     // definición del conjunto de estados de aceptación
// Definición de la función de transición f
f(q_par0,0)= q_impar0
f(q_par0,1)= q_par0
f(q_impar0, 0)= q_par0
f(q_impar0, 1)= q_impar0
```

Figura 2 – Ejemplo de AFD

```
AFN N210 = (Q, \Sigma, \delta, q0, F)
@autor: 'Jeffrey Ullman'
@fecha: 11/04/2023
@version: 1.0.0-alpha
// Definición del conjunto de estados Q
Q= {q0,q1,q2,q3}
\Sigma= {0,1,2}
F= {q3}
\delta(q0,0)= {q0}
\delta(q0,1)= {q0}
\delta(q0,2)= {q0,q1}
\delta(q1,0)=\emptyset
\delta(q1,1)={q2}
\delta(q1,2)=\emptyset
\delta(q2,0)={q3}
\delta(q2,1)=\emptyset
\delta(q2,2)=\emptyset
\delta(q3,0)=\emptyset
\delta(q3,1)=\emptyset
\delta(q3,2)=\emptyset
```

Figura 3 – Ejemplo de AFN

```
AFN-\epsilon Eabc = (Q, \Sigma, f, q0, F)
@autor: ‘John Hopcroft’
@fecha: 11/04/2023
@version: 1.0.0
// Definición del conjunto de estados Q
Q= {q0,q1,q2,q3}
\Sigma= {a,b,c}
F= {q3}
// Definición de la función de transición f
f(q0,a)= {q0}
f(q0,b)= \emptyset
f(q0,c)= \emptyset
f(q0,\epsilon)= {q1}
f(q1,a)= \emptyset
f(q1,b)= {q1}
f(q1,c)= \emptyset
f(q1,\epsilon)= {q2}
f(q2,a)= \emptyset
f(q2,b)= \emptyset
f(q2,c)= {q2}
f(q2,\epsilon)= {q3}
```

f(q3,a)= \emptyset
f(q3,b)= \emptyset
f(q3,c)= \emptyset
f(q3,\epsilon)= \emptyset

Figura 4 – Ejemplo de AFN-ε

Herramientas

- ANTLR (se recomienda versión 4.12.0, disponible en https://www.antlr.org/)
- Editor de texto

Modalidad de Entrega

- Mediante tarea en el Campus Virtual del curso, enviando en un archivo comprimido:
  - Documento en pdf con Carátula, con Nombres y Apellido de los Integrantes y e-mail de cada uno. Agregar un breve comentario sobre la complejidad de la etapa del TP que se entrega y del tiempo utilizado para su resolución. Puede incluir detalles de las decisiones tomadas para resolver lo solicitado.
  - Archivos entregables:
    Primera parte: Archivo `LexerAF`.g4
    Segunda parte*:* Archivos `LexerAFv2`.g4 y `ParserAF`.g4
    Tercera parte*:* Archivos `LexerAFv2`.g4 y `AtributosAF`.g4

Fechas de entrega

Primera parte: 5 de mayo
Segunda parte: 2 de junio
Tercera parte: 16 de junio

Cantidad de integrantes por grupo: 4 alumnos.