Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Organización de Lenguajes y Compiladores 1
Primer Semestre 2020
Ing. Mario Bautista, Ing. Manuel Castillo, Ing. Kevin Lajpop
Aux: Luis Mazariegos, Huriel Gómez, Fernando Paz, Juan Landaverde

Proyecto 1

Objetivos

General

1. Se apliquen los conocimientos sobre los métodos para desarrollar un analizador léxico.

Específico

1. Reforzar los conceptos sobre el método de thompson y el método de cerradura para la transformación de un autómata de tipo AFND a un AFD.
2. Desarrollar el proceso interno de un analizador léxico mediante el uso de los AFD.

Descripción

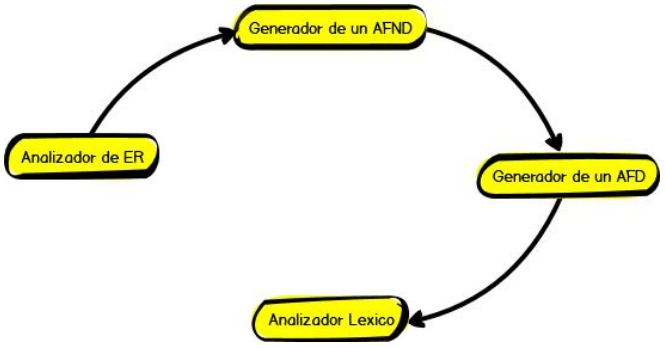
Dado que en la actualidad muchos sistemas de software se ven afectados por los ataques maliciosos para el robo de información, se requiere que como estudiante de ingeniería en sistemas desarrolle una aplicación que realice encriptación de datos, por medio de un sistema que analice cadenas mediante expresiones regulares.

Para dicho sistema se requiere que el análisis cumpla con algunos requerimientos, tales como:

1. Analizador de expresiones regulares: este módulo será el encargado de analizar un archivo de entrada que contendrá expresiones regulares, las cuales leerá una por una.
2. Generador de un AFND y AFD: este módulo creará una tabla de transiciones con la cual se generará el grafo de transiciones para las expresiones regulares este deberá ser implementado mediante el método de thompson + cerradura.
3. Analizador lexico: este módulo realizará la evaluación de cadenas de entrada de manera que identificara los errores léxico contenidos.

Flujo del Programa

La forma en la que se espera que su solución funcione esta dada mediante el siguiente gráfico.



Lenguaje ER

Este interpreta la entrada al momento de la lectura de un archivo de entrada (.er) el cual contiene una o más expresiones regulares que serán reconocidas en el lenguaje.

Definición de Lenguaje

Comentarios
Será permitido el manejo de comentarios de una o más líneas, pueden ser de 2 formas:

- **Comentario de una sola línea:**

```
// Este es un comentario
```

- **Comentarios multilínea**:

```
<!
    Este es un comentario
    multilínea
!>
```

Expresiones Regulares

Las expresiones regulares establecen el patrón que representa al token. Para el análisis y evaluación de cada una de las expresiones regulares se presentarán en notación prefija o polaca.

Descripción de la notación a utilizar:

Notación	Descripción
. a b	Concatenación entre a y b.
| a b	Disyunción entre a y b.
? a	a, 0 o una sola vez.
*** a**	a, 0 o más veces.
+ a	a, 1 o más veces.

Macros o Conjuntos

Un conjunto o macro, son agrupaciones de caracteres del mismo tipo, permitidos en el lenguaje, como agrupaciones de letras, números, etc.
La palabra reservada a utilizar será “CONJ”.

Notación	Descripción
f~j	Conjunto de letras {f,g,h,i,j}.
a~z	Conjunto de letras minúsculas de la ‘a’ a la ‘z’.
A~Z	Conjunto de letras minúsculas de la ‘A’ a la ‘Z’.
0~25	Conjunto de números de 0 a 25.
1,3,5,11,13	Conjunto de números {1,3,5,11,13}.
n,N,e,E	Conjunto de letras {n,N,e,E}.

!~&	Conjunto de signos desde '!' hasta '&' (33 al 38 en código ascii). **Nota:** El rango válido será desde el ascii 32 hasta el 125 omitiendo los ascii de las letras y dígitos.

Notas:

- Un conjunto puede utilizarse dentro de una expresión regular.
- Un conjunto no puede utilizarse en la definición de otro conjunto.

Caracteres especiales del lenguaje:

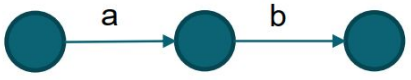
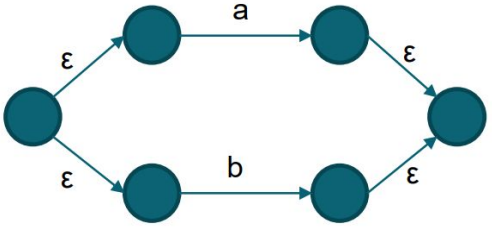
Notación	Descripción
\n	Salto de línea.
\'	Comilla simple.
\"	Comilla doble.
\t	Tabulación.
[:todo:]	Conjunto de cualquier carácter a excepción del salto de línea.

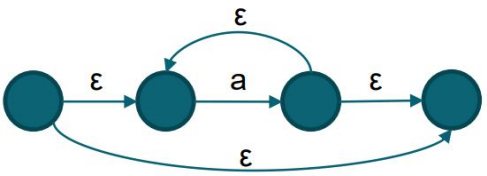
Generación de AFN y AFD

Para la transformación de expresiones regulares reconocidas a Autómatas Finitos Deterministas se hará uso del método de Thompson y cálculo de Cerradura.

Método de Thompson

El método de thompson está basado en la creación de Autómatas Finitos No Deterministas. El mismo se genera por medio de transiciones y estados. Dada una expresión regular con notación prefija, debe transformarse de la siguiente manera.

Notación	Descripción
. a b	
| a b	

*** a**	

Transformaciones en el método de Thompson

Notación	Descripción
? a	**| a ε**
+ a	**. a * a**

Operación Cerradura

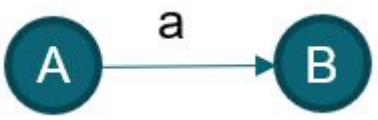
Para la conversión del Autómata Finito No Determinista en uno Determinista se deberán aplicar las siguientes operaciones:

- **Cerradura(X):** Es el conjunto de estados alcanzables desde el conjunto de estados "X" solamente con el uso de transiciones ε.
- **Mover(X,a):** Es el conjunto de estados alcanzables desde el conjunto de estados "X" utilizando transiciones "a".

Procedimiento

1. Se aplica la operación sobre el estado inicial del AFN:
   Cerradura(0)=A
   El conjunto de estados resultantes debe nombrarse, en este ejemplo lo llamaremos el conjunto "A".
2. Debe crearse un nuevo conjunto de estados para cada terminal "a" de la forma:
   Cerradura(Mover(A,a))=B
   Este nuevo conjunto deberá de nombrarse por ejemplo "B". Si ya existe un conjunto de estados idéntico al resultado de la operación se le coloca el mismo nombre.
3. Se repite el paso 2 para cada conjunto de estados que se genere como resultado del mismo. Este paso acaba cuando ya no hay más conjuntos nuevos a los que aplicar el paso 2.
4. Se toman los conjuntos de estados como los nuevos estados del AFD y para cada operación Mover aplicada se utiliza como una transacción en la Tabla de

Transiciones. Cada conjunto de estados que posea al último estado del método de Thompson se convierte en un estado de aceptación.



- El módulo deberá generar tanto el AFN como el AFD para poder mostrarlos por cada lexema analizado. En las gráficas del AFN y AFD se debe mostrar los estados del autómata, el estado inicial, los estados de aceptación y las transiciones del mismo.

Analizador Léxico

Este tiene como finalidad analizar un archivo de entrada que contrenda cadenas (lexema) que deberá evaluar mediante los patrones (expresiones regulares) detectados durante el análisis de expresiones regulares y siempre que se generarán los autómatas finitos deterministas correspondientes a cada expresión regular.

Al terminar el análisis léxico de la cadena de entrada debe generarse una lista de Tokens encontrados o en su defecto una lista de errores en formato XML.

Salida de Tokens

Al momento de terminar el análisis se debe generar una cadena de salida en formato XML el cual contiene los tokens reconocidos. El formato para el archivo XML a desarrollar se ejemplifica a continuación.

```
<ListaTokens>
    <Token>
        <Nombre>nombre_token1</Nombre>
        <Valor>valor_token1</Valor>
        <Fila>fila_token1</Fila>
        <Columna>columna_token1</Columna>
    </Token>
    <Token>
        <Nombre>nombre_token2</Nombre>
        <Valor>valor_token2</Valor>
        <Fila>fila_token2</Fila>
        <Columna>columna_token2</Columna>
    </Token>
</ListaTokens>
```

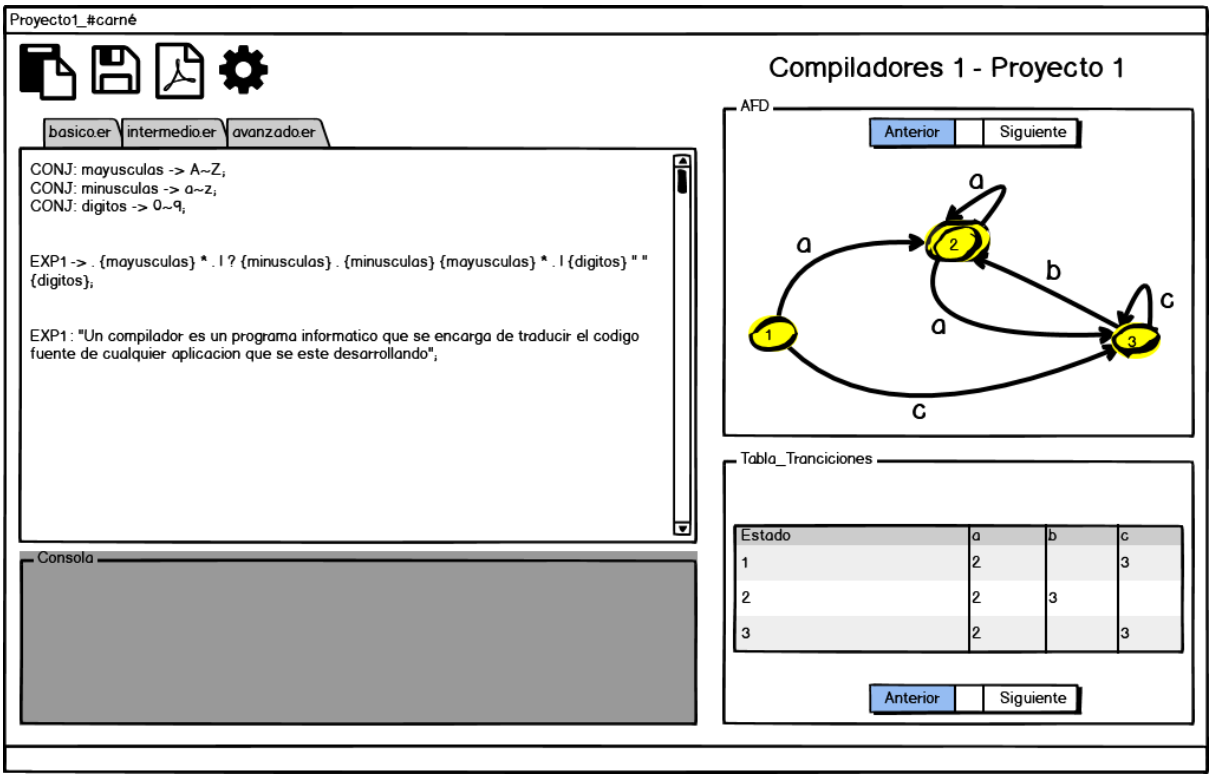
Salida de Errores

En caso de existir errores durante el análisis léxico se debe generarse un archivo XML de errores. El formato para el archivo XML a desarrollar se ejemplifica a continuación.

```
<ListaErrores>
      <Error>
            <Valor>valor_error1</Valor>
            <Fila>fila_error1</Fila>
            <Columna>columna_error1</Columna>
      </Error>
</ListaErrores>
```

Interfaz Gráfica

La aplicación debe contar con una interfaz amigable e intuitiva para que el usuario final interactue con la misma. El diseño de la interfaz gráfica queda a consideración del estudiante pero la facilidad de uso y diseño agradable de la misma se considerarán dentro de la ponderación del proyecto.



La interfaz contará con un editor de texto, un área donde se desplegará el autómata finito determinista y una tabla en donde se podrá ver todas las transiciones obtenidas por el Método de Thompson y Cerradura.

Archivo:

- Abrir: se podrá abrir archivos con extensión ".er"
- Guardar: se podrá guardar el archivo con extensión ".er"
- Agregar Pestaña: se podrán abrir varios archivos ".er" en diferentes pestañas.

Herramientas:

- Cargar Thompson: cargará todas las expresiones regulares utilizando el Método de Thompson.
- Guardar Tokens: después de analizar una cadena se podrán guardar la lista de tokens en extensión ".xml".
- Guardar Errores: después de analizar una cadena y si esta contiene errores se podrán guardar la lista de errores encontrados en extensión ".xml".

Reportes:

- Error léxico: generará un archivo pdf que contendrá una tabla en la cual se podrá ver cada uno de los errores léxicos detectados.

Restricciones

- El Sistema Analítico debe de ser desarrollado en C#, no está permitido el uso de herramientas de ningún tipo para realizar el analizador léxico.
- Deben utilizarse los métodos indicados para realizar la generación de AFD (Thompson y Cerradura), caso contrario no se tendrá derecho a calificación.
- Para las gráficas de Autómatas se utilizará la herramienta Graphviz.
- El proyecto es individual.
- Copias totales o parciales tendrán una nota de 0 puntos y serán reportadas a la Escuela de Ciencias y Sistemas.

Consideraciones

Deben presentarse las funcionalidades que permitan un ciclo de ejecución básico.

Estas son:

- Interfaz gráfica básica para el uso de la aplicación.
- Análisis del archivo ".er".
- Generación de Autómatas Finitos Deterministas utilizando el método de Thompson.
- Reconocimiento de cadenas para la generación de tokens.

Entregables

- Código fuente de la aplicación.
- Aplicación Funcional.
- Repositorio de Versiones.

Fecha de entrega: 26 de marzo de 2020