

MANUAL EXPRESIONES REGULARES/ADF

Elaborado por:

Ricardo Sansores A01105644

Enrique Juárez A00224620

Tomas Cortés A00354103

Índice Contenido

1. Operación de la aplicación-----	3
2. Razón de elaboración del proyecto-----	3
3. Descripción de la aplicación -----	4
4. Componentes del sistema, colaboración entre ellos-----	4
5. Instalación personal-----	7
6. URL's-----	14
7. Generación de autómeta. (Transformar expresión regular a autómeta finito)-----	15
8. Generación de expresión regular (Transformar autómeta finito a expresión regular)-----	20

1. Operación de la aplicación

El sistema se encuentra desarrollado en Java con JSP (Java Server Pages), Sistema JAVA 5.0 SDK.



La herramienta que nos permite obtener el gráfico al para la generación del autómata es una herramienta llamada graphviz. Es un software libre de visualización gráfica, la representación gráfica es un camino para representar la de forma estructurada la información de diagramas y redes. Los gráficos generados en este sistema son en formato "dot" son grafos dirigidos en jerarquía, estos corren como líneas de programa de comando o con una interface gráfica compatible.



El servidor de aplicaciones donde corre la aplicación es Glassfish V2.



2. Razón de elaboración del proyecto

Al comenzar la investigación y validación nosotros como equipo estuvimos definiendo que tipo de aplicación podíamos desarrollar y que aplicará un valor agregado para la misma, decidimos tomar el reto de la transformación de autómatas finitos deterministas a expresiones regulares y viceversa utilizando Graphviz como herramienta modeladora de gráficos.

En primera instancia la elección para el Lenguaje a desarrollo se situaba entre tres tecnologías Java, Python y C++, al final la decisión fue llevarlo a cabo en Java bajo el IDE Netbeans por el expertise de uno de los integrantes del equipo para aprovechar dicha fortaleza en el desarrollo de la aplicación.

La naturaleza del proyecto dejar esta aplicación funcionando al 100% conforme al alcance señalado pero otra de las razones para dejar dicha aplicación es que sea complementada de manera robusta a lo largo del tiempo por las futuras generaciones que tomen esta clase en lugar de reiniciar un proceso de trabajo final desde cero.

3. Descripción del Proyecto

Nuestro Proyecto se encuentra desarrollado en tecnología Java, JSP (Java Server Pages), Graphviz como entorno que nos permite obtener los grafos para la generación de las imágenes transformadas de formato **dot a formato PNG**. Esto a lo concierne a la transformación de expresión regular-autómata Finito.

En lo se refiere a la transformación de un autómata finito convertido a expresión regular por medio del Conversor de expresiones regulares infix a postfix y el Objeto Autómata Finito No determinista lo que nos permite tener como resultado la expresión regular.

Las imágenes son almacenadas en la carpeta **res** al igual que el formato dot generado por graphviz.

La aplicación es una aplicación WEB escrita HTML, JSP y JQUERY para ingresar los datos requeridos al momento de la generación del autómata.

El servidor utilizado es Glassfish V2 se utilizó dicho Server puesto que la versión 3 aun cuenta con áreas de oportunidad y se utilizó JDK 5 ya que el JDK 7 al igual que el Server tiene ventanas de oportunidad para que mantenga una ejecución sin fallas.

4. Componentes del sistema.

1. Conversor de expresiones regulares infix a postfix.

Para facilitar el parseo de la expresión regular para convertirla posteriormente en un autómata finito en memoria principal se toma una expresión regular en formato infix y se transforma en una expresión regular en formato postfix. Esté componente únicamente soporta por el momento operadores de unión, concatenación y cerradura de klenee. Todo símbolo que no sea un operador es considerado un caracter.

2. Objeto Autómata Finito No determinista.

Es un objeto que modela a todo el autómata utilizando objetos "Nodo" y objetos "Transición". El objeto se puede construir utilizando una expresión regular en formato postfix o utilizando una serie de arreglos de strings que representan cada uno de los nodos del autómata así como sus transiciones, estado inicial y estados de aceptación.

3. Dot Helper

Una vez que se tiene el autómata almacenado en memoria principal en forma de grafo se envía al objeto DotHelper que es una utilidad que creamos que toma un objeto

"AutomataFinitoNoDeterminista" y lo traduce a formato DOT. El formato DOT es un DSL (Domain Specific Language) que es utilizado para representar gráficas en archivos de texto. Una vez que se tiene el archivo de texto se envía al controlador "ConverterMB" para que sea transformado utilizando la herramienta graphviz que comprende el formato DOT a una imagen en formato PNG.

4. ConverterMB

Es un controlador que realiza principalmente 3 tareas. Su primera tarea es recibir los autómatas y expresiones regulares generadas por el usuario en el browser y comunicárselos al resto de los componentes de la aplicación. Su segunda tarea es tomar los autómatas y expresiones resultantes enviarlas al browser el usuario. Finalmente es el intermediario entre el componente DotHelper y el browser ya que toma los archivos "DOT" generados por el componente "DotHelper" y los envía a Graphviz para generar imágenes PNG que son presentadas en el browser del cliente.

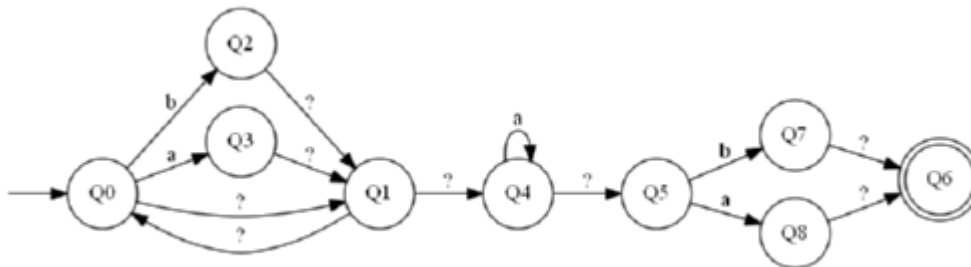
5. Vista

Es una página WEB escrita en HTML, JSP y JQUERY que le brinda al usuario componentes para capturar expresiones regulares, autómatas finitos y ver los resultados de sus peticiones.

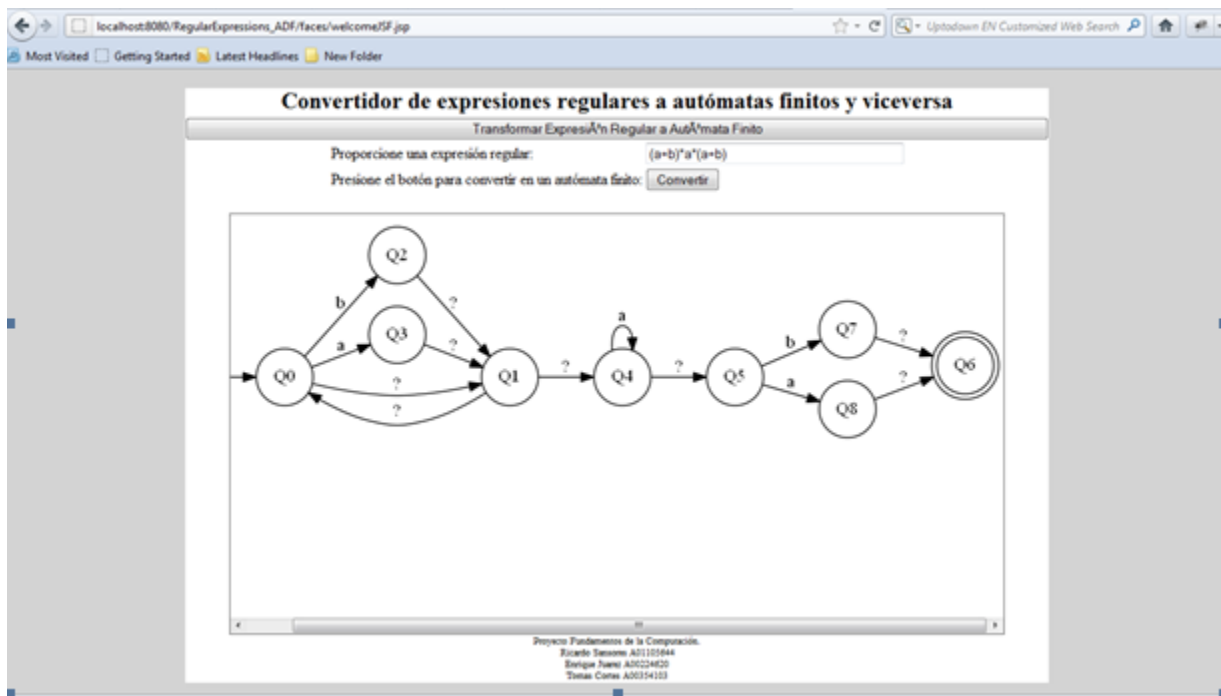
Las imágenes de los autómatas son almacenadas en la carpeta **res** al igual que los archivos dot como lo señala el componente dot helper se almacena en memoria principal el grafo y lo transforma en formato dot como de la siguiente manera:

```
digraph finite_state_machine {
    rankdir=LR;
    size="42,42"
    EMPTY [style=invis]    EMPTY [shape=point]    node [shape =
doublecircle]; Q6;
    node [shape = circle];
    EMPTY -> Q0 [ label = "" ];
    Q0 -> Q2 [ label = "b" ];
    Q0 -> Q3 [ label = "a" ];
    Q0 -> Q1 [ label = "?" ];
    Q1 -> Q0 [ label = "?" ];
    Q1 -> Q4 [ label = "?" ];
    Q2 -> Q1 [ label = "?" ];
    Q3 -> Q1 [ label = "?" ];
    Q4 -> Q4 [ label = "a" ];
    Q4 -> Q5 [ label = "?" ];
    Q5 -> Q7 [ label = "b" ];
    Q5 -> Q8 [ label = "a" ];
    Q7 -> Q6 [ label = "?" ];
    Q8 -> Q6 [ label = "?" ];
}
```

Enseguida el formato dot es tomado por el componente convertir MB desde el dot helper envía al Graphviz y generando la imagen en formato PNG y esta es enviada al browser del cliente.



La imagen PNG es llevada al browser del cliente.



5. instalación Personal

Instalación de Glassfish

Antes de instalar el servidor GlassFish, hay que tener en el ordenador **correctamente instalado el JDK**.

- 1.- Ve a http://java.sun.com/javase/downloads/index_jdk5.jsp
- 2.- Selecciona JDK 5.0 Update 16 download.
- 3.- Decide cual Platform (Windows) y marca el checkbox de los términos de uso.
- 4.- Selecciona jdk-1_5_0_16-windows-i586-p-iftw.exe para instalar el software.
- 5.- Instala el JDK.

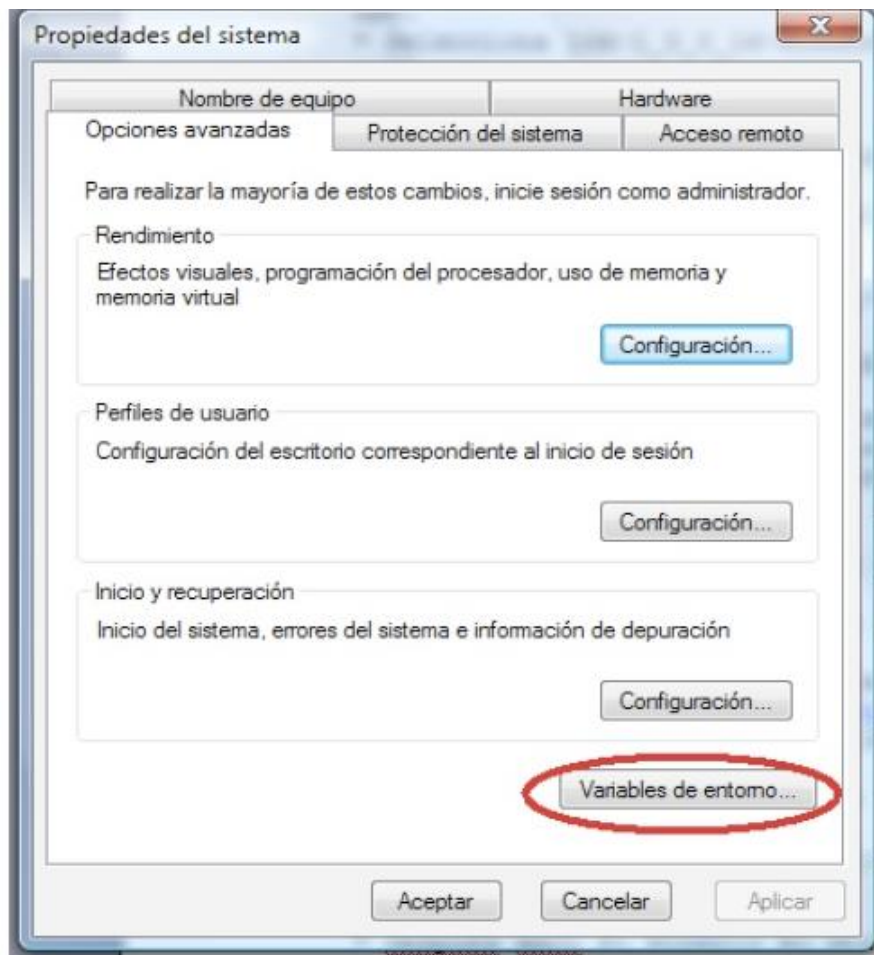
Por defecto se instala en (aunque se puede instalar donde se desee):

C:\Archivos de programa\java\jdk1.5.0_16

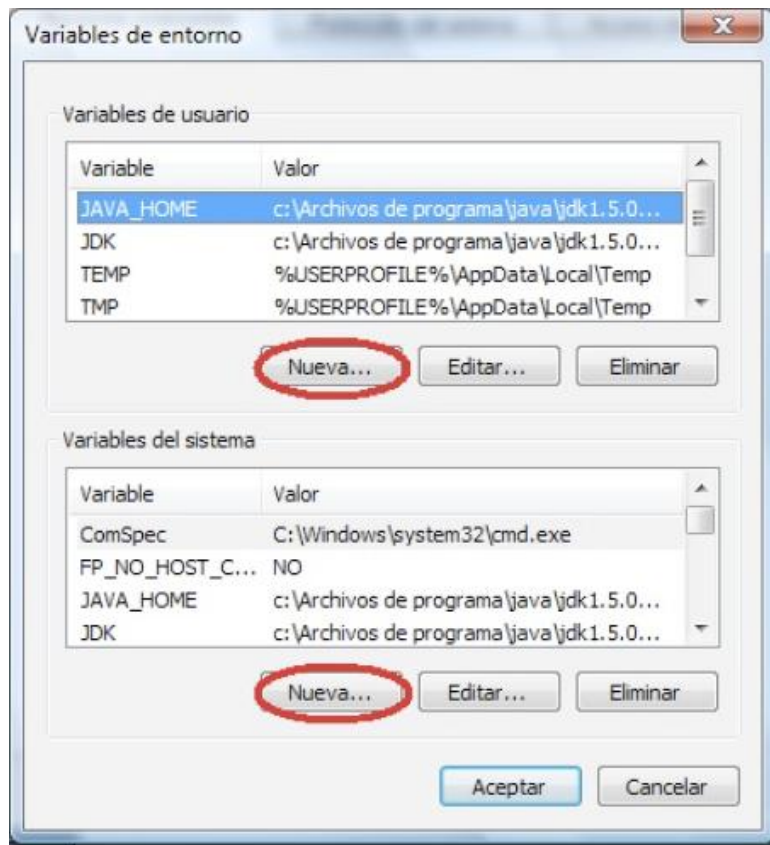
El próximo paso, crear la variable JAVA_HOME.

La forma es la siguiente:

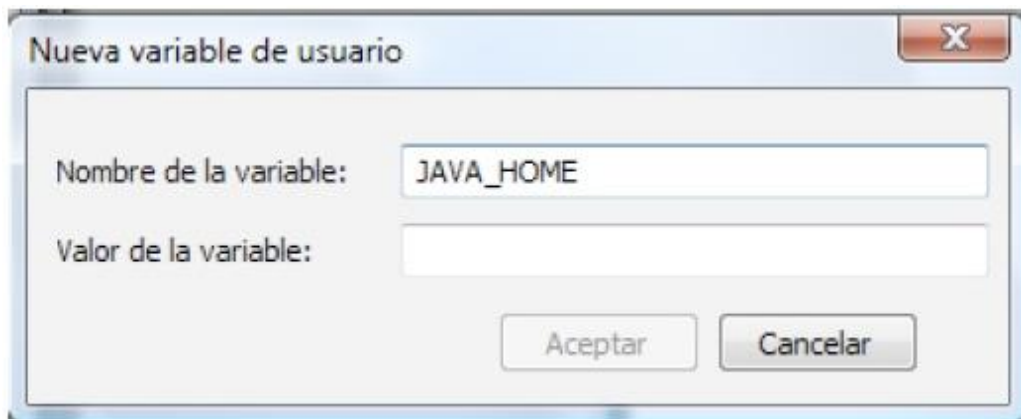
- 1.- Abrir el Panel de control
- 2.- Seleccione "Ver Vista clásica"
- 3.- Seleccione icono Sistema
- 4.- En el panel de Sistema de Propiedades, seleccione "Configuración Avanzada del sistema"
- 5.- Después en la pestaña "Opciones Avanzadas" seleccionar Variables de Entorno.



6.- En el panel de Variables de entorno, seleccione Nuevo

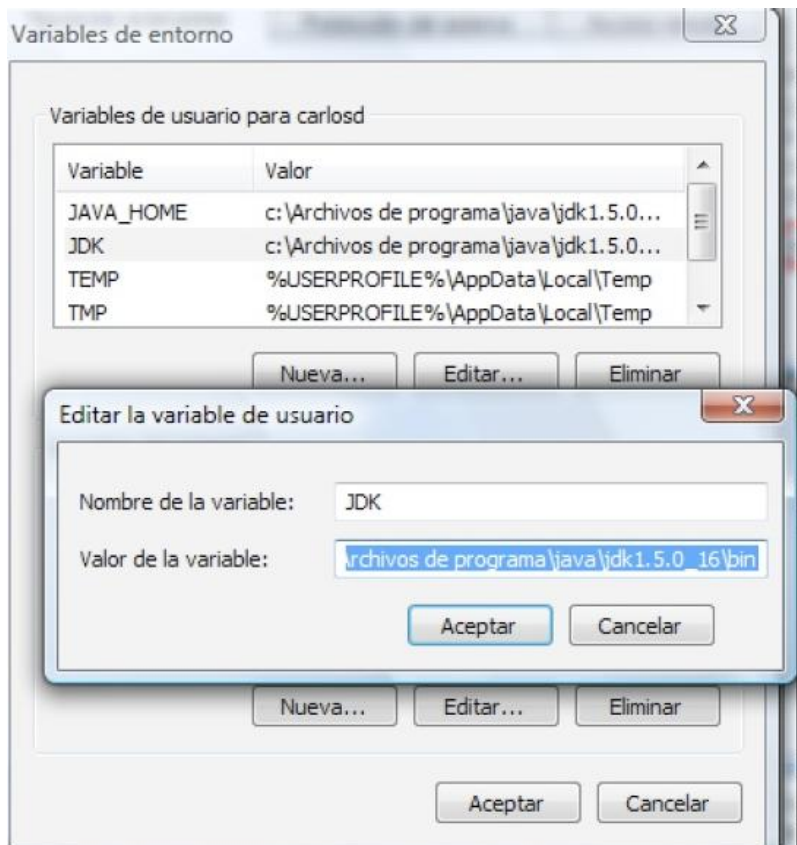


7.- En el panel de Nueva Variable de Usuario, escribe en Nombre de la Variable JAVA_HOME.



8.- Después en el Valor de la Variable, agregar en donde se instalo el JDK. Ejemplo: C:\\Archivos de programa\\java\\jdk1.5.0_16

9.- Añade en el Path variable los binarios del JDK. Ejemplo: C:\\Archivos de programa\\java\\jdk1.5.0_16



10.- Presionar Aceptar en el panel de Variables de Entorno y Propiedades del Sistema.

En teoría, ya está. Una vez en este punto nos dirigimos a instalar el GlassFish v2.1:

1.- Tienes que bajar el binario a tu computadora. Baje el binario: "glassfish-installer-v2.1-b50-windows.jar". Es de 54MB.

2.- Después mover el binario al directorio (drive) C:

3.- Usando la herramienta Command Prompt "Consola de Símbolo del Sistema", ejecute la instalación acuerdo a los pasos:

- Escribir en el directorio C:\\ java -Xmx256 -jar glassfish-installer-v2.1-b50-windows-jar

```
Símbolo del sistema
Domain [domain1] is running [Sun Java System Application Server 9
-fcs] with its configuration and logs at: [C:\glassfish\domains1
Admin Console is available at [http://localhost:4848].
Use the same port [4848] for "asadmin" commands.
User web applications are available at these URLs:
[http://localhost:8080 https://localhost:8181 ].
Following web-contexts are available:
[/web1 /__wstx-services ].
Standard JMX Clients (like JConsole) can connect to JMXServiceURL
[service:jmx:rmi:///jndi/rmi://OBP15:8686/jmxrmi] for domain manag
Domain listens on at least following ports for connections:
[8080 8181 4848 3700 3820 3920 8686 ].
Domain does not support application server clusters and other sta
es.

C:\glassfish\bin>cd..
C:\glassfish>cd..
C:\>java -Xmx256m -jar glassfish-installer-v2.1-b50-windows.jar
```

4.- Siguiente, aparece un panel de licencia. Desplaza el Scroll hasta el fin del panel y Acepta los términos.

5.- El instalador se abre e instala el software en un directorio llamado "glassfish" lo instala en C:/glassfish

- Teclee cd glassfish

```
Símbolo del sistema
Admin Console is available at [http://localhost:4848].
Use the same port [4848] for "asadmin" commands.
User web applications are available at these URLs:
[http://localhost:8080 https://localhost:8181 ].
Following web-contexts are available:
[/web1 /__wstx-services ].
Standard JMX Clients (like JConsole) can connect to JMXServiceURL
[service:jmx:rmi:///jndi/rmi://OBP15:8686/jmxrmi] for domain manag
Domain listens on at least following ports for connections:
[8080 8181 4848 3700 3820 3920 8686 ].
Domain does not support application server clusters and other sta
es.

C:\glassfish\bin>cd..
C:\glassfish>cd..
C:\>cd glassfish
C:\glassfish>lib\ant\bin\ant -f setup.xml
```

6.- En el directorio glassfish ingrese "lib/ant/bin/ant -f setup.xml". Esto sirve para correr la instalación.

Iniciar el servidor a partir del dominio. Un dominio proporciona la autenticación y la administración para el servidor. A partir del dominio se inicia la instancia del servidor en el dominio. Cuando se instala el servidor GlassFish se crea un dominio predeterminado llamado domain1.

1. Ingrese C:\glassfish\bin> asadmin start-domain domain1

```
C:\> Símbolo del sistema
30/10/2008 10:41 a.m.      865 wscompile.bat
30/10/2008 10:41 a.m.      865 wsdeploy.bat
30/10/2008 10:41 a.m.      842 wsgen.bat
30/10/2008 10:41 a.m.      857 wsimport.bat
30/10/2008 10:41 a.m.      818 xjc.bat
          16 archivos      41,789 bytes
          2 dirs  48,553,074,688 bytes libres

C:\glassfish\bin>asadmin start-domain domain1
```

Nota: Cuando el servidor se ha iniciado, aparece este mensaje: domain1 dominio está listo para recibir solicitudes de clientes.

Instalación Glassfish en Linux

1. Instalamos JDK,
2. Descargamos Glassfish.
3. Copiamos a /opt [cp glassfish-installer-v2.1.1-b31g-linux.jar /opt]
4. Separamos el GlassFish y creamos una nueva estructura de directorios raíz en un directorio llamado “glassfish” con el comando

```
[java -Xmx256m -jar glassfish-installer-v2.1.1-b31g-linux.jar]
```

5. Ingresamos a la nueva carpeta [cd glassfish]
6. Agregamos permisos de ejecución [chmod -R +x lib/ant/bin]
7. Iniciamos el setup [lib/ant/bin/ant -f setup.xml]

Instalación Graphviz

Para instalar graphviz, basta con poner como root en un terminal y aceptar.

#apt-get install graphviz graphviz-dev graphviz-doc

Esto instala entre otras cosas, la aplicación dot que nos permitirá convertir un archivo .dot a un archivo .ps (PostScript) o .png (Portable Network Graphic) con el dibujo del grafo y las páginas del manual con el detalle de la sintaxis para dot.

Los archivos .dot tienen la siguiente sintaxis.

```
/*Esto es un comentario*/
graph nombre_del_grafo {
    "idNodo1";
    "idNodo2"; /*estos son identificadores de nodos*/
    "idNodo3";
    "idNodo1" -- "idNodo2";
    "idNodo1" -- "idNodo3"; /*estas son relaciones entre nodos*/
}
```

Instalación Graphviz

1. Descarga Graphviz source package:
Ve http://www.graphviz.org/Download_source.php y copialo sobre la versión estable graphviz-*VERSION*.tar.gz o el recurso de desarrollo graphviz-working.tar.gz.
2. Descomprime el paquete:

3. Se puede usar,

```
gunzip < graphviz-working.tar.gz | tar xf -
```

Vamos a \$ROOT el directorio en donde se encuentra almacenado el recurso de Graphviz.

4. Configure el paquete en tu sistema.

```
cd $ROOT
configure --disable-shared --enable-static --with-mylibgd --disable-
swig --without-x --without-tclsh --with-codegens --disable-ltdl
```

6. URL's

URL Principal

http://www.sicom.mx:8100/RegularExpressions_ADF

URL BACKUP

http://216.154.208.156:8100/RegularExpressions_ADF/faces/welcomeJSF.jsp

SVN Googlecode

<http://code.google.com/p/regular-expressios-to-adf/>

1. Generación de autómeta.(Transformar expresión regular a autómeta finito).

Transformar Expresión Regular a Autómeta Finito

Así luce nuestra pantalla principal para la generación de un autómeta finito.

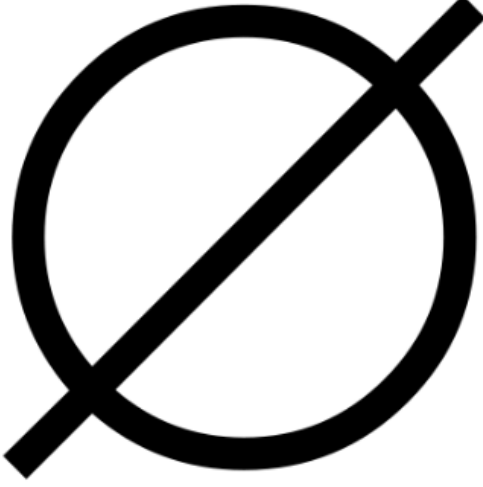
Convertidor de expresiones regulares a autómetas finitos y viceversa

Transformar Autómeta Finito a Expresión Regular

Proporcione el alfabeto a utilizar: ?

Proporcione los estados a utilizar: ?

Resultado: Conjunto Vacio



Proyecto Fundamentos de la Computación.
Ricardo Sansores A01105644
Enrique Juarez A00224620
Tomas Cortes A00354103

Enseguida mostraremos paso a paso las instrucciones para la generación de un autómeta finito.

Proporcione el alfabeto sobre el cual realizará operará el autómeta. Está premitido cualquier letra y número. El alfabeto reconoce como caracteres diferentes mayúsculas y minúsculas y es importante no usar caracteres de control o símbolos (!, ¡, ¿, +, etc.) ya que pueden entrar en conflicto con los caracteres de control de la expresión regular. Cada uno de los caracteres del alfabeto debe encontrarse separado por una coma.

Convertidor de expresiones regulares a autómatas finitos y viceversa

Transformar Autómata Finito a Expresión Regular

Proporcione el alfabeto a utilizar: ?

Proporcione los estados a utilizar: ?

Proporcione los estados que usará su autómata finito. Debe separar cada estado del autómata con comas y solo acepta estados de un solo carácter ya que no aceptará estados de dos o más caracteres como ejemplo q2 , xy entre otros.

Proporcione los estados a utilizar: ?

Así luce los espacios una vez ingresados los datos necesarios para la generación de nuestra tabla.

Convertidor de expresiones regulares a autómatas finitos y viceversa




Transformar Autómata Finito a Expresión Regular

Proporcione el alfabeto a utilizar: ?

Proporcione los estados a utilizar: ?




Una vez teniendo los datos como en la imagen anterior de manera correcta, presionamos click sobre el botón crear tabla,

Se generará en la parte de abajo donde teníamos como resultado el conjunto vacío, puesto que no había ningún dato previo ese es el resultado que desplegaba, se desplegará una tabla como se muestra en la imagen para comenzar a ingresar en los campos las transiciones de un estado a otro.

Cambiar estado inicial:   




	a	b
★ 1	<input type="text"/>	<input type="text"/>
2	<input type="text"/>	<input type="text"/>
3	<input type="text"/>	<input type="text"/>

Inserte en cada celda el estado al que se realizará la transición al recibir el caracter marcado en la columna correspondiente.


Cambiar estado inicial:   




	a	b
★ 1	2 <input type="text"/>	3 <input type="text"/>
2	2 <input type="text"/>	2 <input type="text"/>
3	1 <input type="text"/>	2 <input type="text"/>


Puede cambiar el estado utilizando los botones de arriba y abajo representados como el objeto de flechas azules donde donde está la leyenda “cambiar estado inicial” notaremos que cambiamos de estado inicial ya que el objeto de estrella se trasladará hacia el lugar siguiente.

Cambiar estado inicial:   

	a	b
1	2 <input type="text"/>	3 <input type="text"/>
★ 2	2 <input type="text"/>	2 <input type="text"/>
3	1 <input type="text"/>	2 <input type="text"/>

Puede indicar si un estado es de aceptación presionando sobre la etiqueta del estado en esta caso presionamos el estado 3 como estado de aceptación, como prueba de que hemos hecho esta operación de manera correcta se desplegará una palomita  3 al lado izquierdo del estado de aceptación marcado,

Cambiar estado inicial:   

	a	b
★ 1	2	3
2	2	2
 3	1	2

Al concluir la edición presione el botón convertir.

Y obtendremos el resultado en la parte del espacio de nuestra tabla.

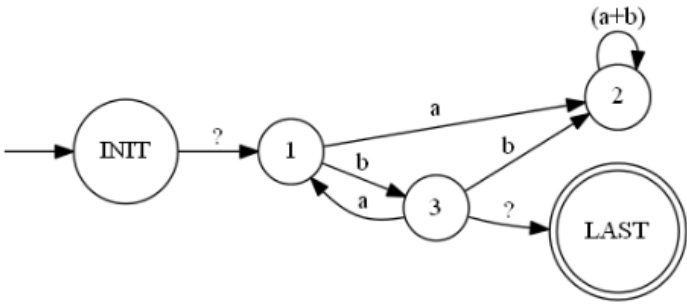
Convertidor de expresiones regulares a autómatas finitos y viceversa

Transformar Autómatas Finitos a Expresión Regular

Proporcione el alfabeto a utilizar:

Proporcione los estados a utilizar:

Resultado: $((sb)(ab)^*\epsilon)$



Proyecto Fundamentos de la Computación.
Ricardo Sansores A01105644
Enrique Juárez A00224620
Tomas Cortes A00354103

Como resultado obtenemos la generación del autómata gráficamente y además la generación del estado de nuestra expresión regular en la leyenda “resultado”

Resultado: $((\epsilon b)(ab)^*\epsilon)$

Con esto concluimos la primera parte de nuestro programa transformación de expresión regular a autómata finito.

Para cambiar de pantalla o continuar con nuestro siguiente paso es de transformar autómata finito a expresión regular presionamos el botón siguiente

Transformar Autómata Finito a Expresión Regular

Se encuentra situado en la parte superior enseguida del título de nuestro sistema.

Convertidor de expresiones regulares a autómatas finitos y viceversa

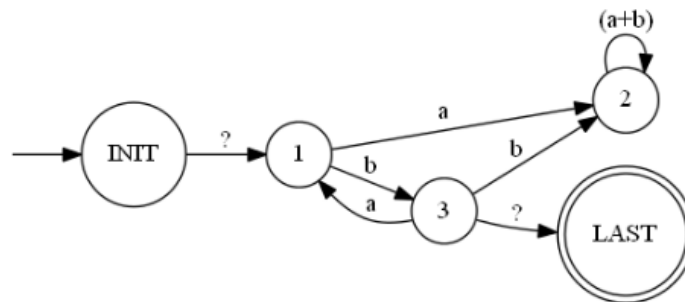
Transformar Autómata Finito a Expresión Regular

Proporcione el alfabeto a utilizar:

Proporcione los estados a utilizar:

Crear Tabla

Resultado: $((\epsilon b)(ab)^*\epsilon)$



Proyecto Fundamentos de la Computación.
Ricardo Sansores A01105644
Enrique Juárez A00224620
Tomas Cortes A00354103

2. Generación de expresión regular (Transformar autómata finito a expresión regular).

Transformar Autómata Finito a Expresión Regular

Al presionar este botón nos conduce a la siguiente pantalla

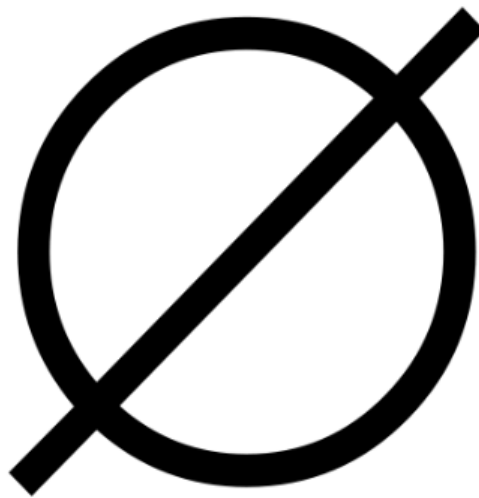
Convertidor de expresiones regulares a autómatas finitos y viceversa

Transformar Expresión Regular a Autómata Finito

Proporcione una expresión regular:

Presione el botón para convertir en un autómata finito:

Convertir



Proyecto Fundamentos de la Computación.
Ricardo Sansores A01105644
Enrique Juárez A00224620
Tomas Cortes A00354103

Es muy similar a la pantalla inicial pero en esta solo proporcionaremos una expresión regular en el campo donde nos lo requiere.

Proporcione una expresión regular:

Presione el botón para convertir en un autómata finito:

Convertir

Ingresamos nuestra expresión regular a transformar en este caso es: $(a+b)^*(a+b)$

Los caracteres a ingresar deben ser ingresados sin espacios.

Proporcione una expresión regular:

Presione el botón para convertir en un autómata finito:

Presionamos el botón convertir

Presione el botón para convertir en un autómata finito:

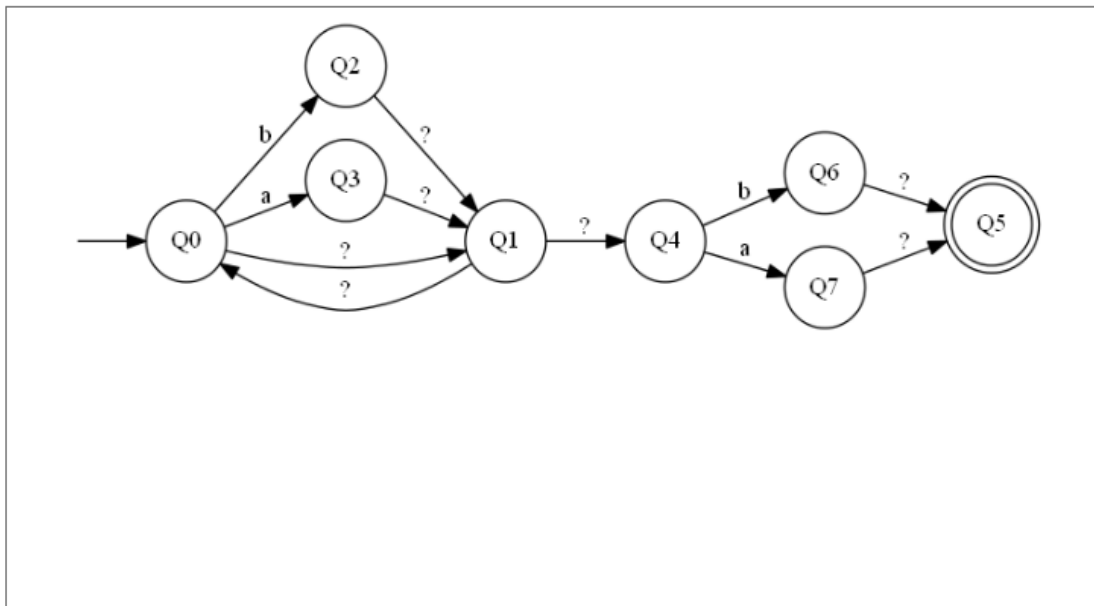
Y en nuestra pantalla se desplegará el resultado como lo muestra la imagen.

Convertidor de expresiones regulares a autómatas finitos y viceversa

Transformar Expresión Regular a Autómata Finito

Proporcione una expresión regular:

Presione el botón para convertir en un autómata finito:



Proyecto Fundamentos de la Computación.
Ricardo Sansores A01105644
Enrique Juárez A00224620
Tomas Cortes A00354103

Y así consecutivamente podemos ingresar expresiones regulares para ser transformadas en autómata finito.