Universidad Tecnológica de la Mixteca

Ingeniería en Computación

Compiladores

Práctica:

Reporte de la Implementación del algoritmo de Construcción de Conjuntos

Profesor:

Ing. David Martínez Torres

Grupo:

502-A

Equipo:

3

Integrantes del equipo:

Alan Yael Lopez Rojas
Elorza Velásquez Jorge Aurelio
García Ramírez Fernando
Martinez Lorenzo Frida Ximena
Ángel de Jesús Alvarado Torres
José Guadalupe Robles Jiménez

Fecha de entrega:

26 / 10 / 2023

Introducción:

La implementación del algoritmo de Construcción de Conjuntos es un proceso fundamental en la teoría de la computación y la automatización. Su objetivo principal es convertir un Autómata Finito No Determinista (AFN) en un Autómata Finito Determinista (AFD), lo que simplifica el análisis y procesamiento de lenguajes formales.

Este algoritmo toma como entrada un Autómata Finito No Determinista (AFN) o una expresión regular, desde la cual se construye el AFN. La entrada se proporciona a través de un archivo que contiene la descripción del autómata o la expresión regular.

La conversión de un AFN a un AFD es un paso importante en la teoría de la computación y tiene aplicaciones en el diseño de compiladores, análisis léxico, y en la resolución de problemas que involucran el reconocimiento de patrones y cadenas de texto. El algoritmo de Construcción de Conjuntos es una herramienta esencial en este contexto, y su implementación es crucial para automatizar procesos que dependen de la manipulación de lenguajes formales.

Desarrollo:

El algoritmo de Construcción de Conjuntos es una técnica fundamental en la teoría de autómatas y lenguajes formales. Su principal objetivo es convertir un Autómata Finito No Determinista (AFN) en un Autómata Finito Determinista (AFD). Aquí, se destaca una breve reseña del algoritmo y sus posibles funcionalidades, así como los requisitos clave para programarlo dentro de nuestro Backlog:

Funcionalidades Clave:

- **Conversión de AFN a AFD:** La función principal del algoritmo es tomar un AFN como entrada y generar un AFD equivalente. Esto simplifica el procesamiento de lenguajes regulares y facilita la implementación de sistemas de reconocimiento de patrones y cadenas de texto.

- **Tabla de Transiciones:** El algoritmo genera una tabla de transiciones para el AFD resultante. Esta tabla es esencial para comprender y representar las relaciones entre los estados del autómata y las transiciones de entrada.

- **Identificación de Estados Equivalentes:** Durante la construcción del AFD, el algoritmo identifica y combina estados equivalentes, reduciendo así el número de estados y simplificando la representación del autómata.

- **Reconocimiento de Cadenas:** El AFD construido se puede utilizar para reconocer si una cadena de entrada pertenece al lenguaje aceptado por el autómata.

Requisitos:

Entrada del AFN: Se necesita una representación adecuada del AFN de entrada. Esto podría ser en forma de una estructura de datos que almacene los estados, el alfabeto, las transiciones, el estado inicial y los estados de aceptación.

Tabla de Transiciones: La capacidad de construir y mantener una tabla de transiciones es fundamental. Esto implica un conocimiento sólido de estructuras de datos y la manipulación eficiente de las transiciones entre estados.

Algoritmo de Conjuntos: La implementación del algoritmo de Construcción de Conjuntos en sí mismo es esencial. Esto requiere una comprensión profunda del algoritmo y su lógica.

Interfaz de Usuario: Para hacer la herramienta accesible, se puede desarrollar una interfaz de usuario que permita a los usuarios cargar un AFN, ejecutar el algoritmo y visualizar el AFD resultante, así como la tabla de transiciones.

Manejo de Errores: Es importante incorporar manejo de errores y validación para garantizar que el AFN de entrada sea válido y que el proceso de conversión sea robusto.

Pruebas y Verificación: Realizar pruebas exhaustivas para verificar la corrección de la implementación es esencial. También, se pueden incluir pruebas de rendimiento para evaluar la eficiencia del algoritmo en casos de uso real.

En resumen, la implementación del algoritmo de Construcción de Conjuntos es un proyecto que requiere un conocimiento profundo de la teoría de los autómatas. Su capacidad para convertir AFNs en AFDs es de gran utilidad en aplicaciones que involucran el procesamiento de lenguajes regulares, como análisis léxico en compiladores y reconocimiento de patrones. Una vez teniendo esto en claro comenzamos con nuestro cronograma para poder avanzar con el proyecto.

		Jorge	***Fernando***	***Frida***	***Ángel***	***Alan***	***José***
22 de octubre	*Lo que hice hoy*	Hice un pequeño tutorial paso a paso para explicar el algoritmo de teoría de conjuntos	Se hizo investigación para la tercera entrega del trabajo	Se detectó un error en el algoritmo de Thompson. Y se comenzó con el rastreo, se apuntaba a la cerradura de Kleene.	Se empezó a trabajar en el algoritmo de conjuntos, en espera de la función de la cerradura-E	Se realizó la interfaz gráfica del algoritmo de conjuntos	Se realizó el algoritmo de cerradura-e y se probó para el primer caso,además se trabaja en el caso general
	Contratiempos	–	-	-	-	-	-
23 de octubre	*Lo que hice hoy*	Se comenzó a plantear algunos requerimientos para el analizador léxico y se hicieron pruebas al algoritmo de Thompson para corregir bugs	Se empezó a generar ideas para la implementación del analizador léxico	Se comenzó con la investigación para el desarrollo del análisis léxico.	Se unió el algoritmo de conjuntos con la función de la cerradura y se corrigieron algunos errores, aun faltan algunos por corregir.	Se implementó el algoritmo de la cerradura-e	Se implementó el algoritmo de la cerradura-e
	Contratiempos	–	-	Trabajé en mi proyecto de arquitectura y en el desarrollo de mi videojuego para Programación Web.	-	-	-

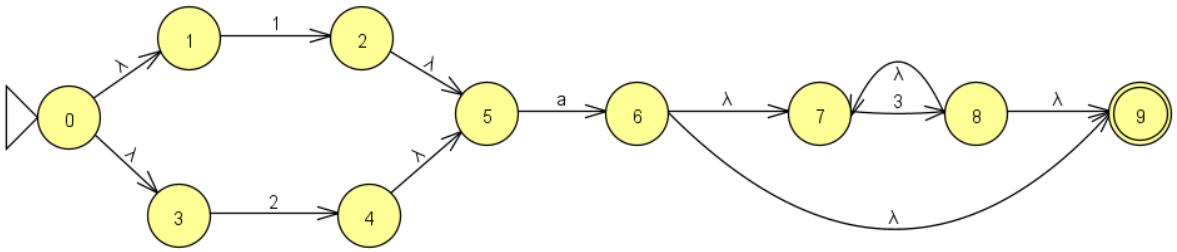
24 de octubre	*Lo que hice hoy*	Hice algunas pruebas de AFN y expresiones regulares para probar algunos casos sencillos para el programa	Se pensaron formas para manejar los errores para la siguiente etapa del analizador léxico	Se rastreó y corrigió el error existente en el algoritmo de Thompson, que impedía una concatenación correcta con las cerraduras.	Se empezó con la implementación de la nueva ventana (Construcción de Conjuntos)	Se implementaron mejoras y corrigieron fallas en los 2 algoritmos pero hace falta renombrar los estados	Se implementaron mejoras y corrigieron fallas en los 2 algoritmos pero hace falta renombrar los estados
	Contratiempos	No se pasaron en limpio las pruebas aún para la revisión de todo el equipo	–	–	–	Genera basura en las transiciones	–
25 de octubre	*Lo que hice hoy*	Se comenzó a crear un archivo para detectar la tira de tokens del analizador léxico	Se continuó con las ideas para la definición del AFD a usarse en el analizador léxico, que sea reconocible por el programa	Se añadió la marca de estado final e inicial al momento de imprimir los datos en las tablas mostradas en la interfaz	Se terminó la nueva ventana procesando los nuevos estados ya renombrados	Se finalizó el algoritmo de construcción de conjuntos para casos sin palabras reservadas	El algoritmo se terminó de construir para casos generales sin que la expresión lleve if o for.
	Contratiempos	–	Debo tener un mejor conocimiento sobre el funcionamiento del programa final	–	–	–	Aún no funciona para casos con if o for.

26 de octubre	*Lo que hice hoy*	Se hizo un test del programa y se depuró, también se unió el documento para el reporte de construcción de conjuntos	Se trabajó en el diseño de la tercera implementación, incluyendo la lectura de datos, y su compatibilidad con el código existente	Se comenzó a trabajar sobre el diseño e implementación de la tercer entrega, creación de una tabla hash para el procesamiento de los token	Se finalizó la programación del algoritmo de construcción de conjuntos para casos más complejos con las palabras dígitos y letras	Se finalizó la programación del algoritmo de construcción de conjuntos.	Se finalizó la programación del algoritmo, corrigiendo algunos errores sobre la impresión de las transiciones.
	Contratiempos	–	–	–	–	-	Tuvimos algunos problemas para imprimir la salida pero se corrigieron.

Para este último día, además como se menciona se unió la documentación, que es producto de nuestras reuniones diarias por la tarde, mientras que en la mañana en las horas de implementación se avanzaba solucionado bugs y depurando el código al atardecer se daban conclusiones del trabajo individual como se mostró.

En la siguiente página se añaden algunas de las pruebas hechas para el algoritmo de construcción de conjuntos

Antiguo AFN:



$s = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ $s_0 = \{0\}$

$\Sigma = \{1, 2, 3, a\}$ $F = \{9\}$

$RegExp = (1|2)a(3\ *)$

$c - \varepsilon(0) = \{0, 1, 3\}$

$A = \{0, 1, 3\}$

$c - \varepsilon((mueve(A, 1)) = c - \varepsilon(2) = \{2, 5\}$ $B = \{2, 5\}$

$c - \varepsilon((mueve(A, 2)) = c - \varepsilon(4) = \{4, 5\}$ $C = \{4, 5\}$

$c - \varepsilon((mueve(B, a)) = c - \varepsilon(6) = \{6, 9, 7\}$ $D = \{6, 7, 9\}$

$c - \varepsilon((mueve(C, a)) = c - \varepsilon(6) = \{6, 9, 7\}$ $D = \{6, 7, 9\}$

$c - \varepsilon((mueve(D, 3)) = c - \varepsilon(8) = \{8, 7, 9\}$ $E = \{7, 8, 9\}$

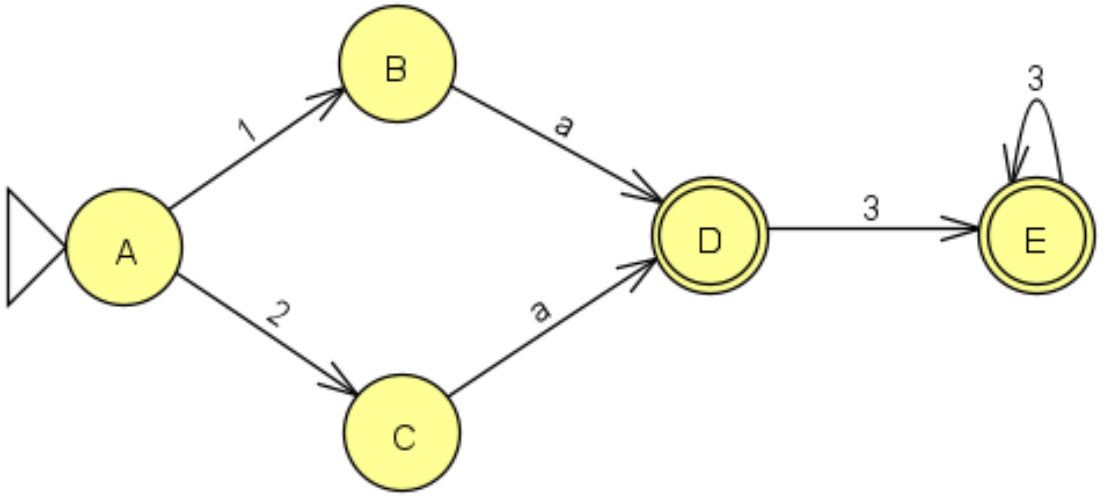
$c - \varepsilon((mueve(E, 3)) = c - \varepsilon(8) = \{8, 7, 9\}$ $E = \{7, 8, 9\}$

Nuevo AFD:

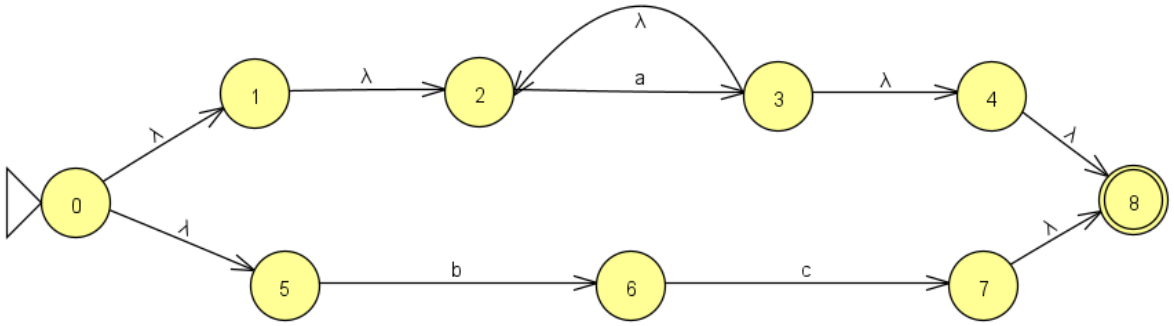
$s = \{A, B, C, D, E\}$ $s_0 = \{A\}$

$\Sigma = \{1, 2, 3, a\}$ $F = \{D, E\}$

Conjunto de estados AFD	1	2	a	D
$\rightarrow A$	B	C	-	-
B	-	-	D	-
C	-	-	D	-
D_F	-	-	-	E
E_F	-	-	-	E



Antiguo AFN:



$s = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ $s_0 = \{0\}$

$\Sigma = \{a, b, c\}$ $F = \{8\}$

$RegExp = (a\ +)|(bc)$

$c - \varepsilon(0) = \{0, 1, 5, 2\}$

$A = \{0, 1, 2, 5\}$

$c - \varepsilon((mueve(A, a)) = c - \varepsilon(3) = \{3, 4, 8, 2\}$ $B = \{2, 3, 4, 8\}$

$c - \varepsilon((mueve(A, b)) = c - \varepsilon(6) = \{6\}$ $C = \{6\}$

$c - \varepsilon((mueve(B, a)) = c - \varepsilon(3) = \{3, 4, 8, 2\}$ $B = \{2, 3, 4, 8\}$

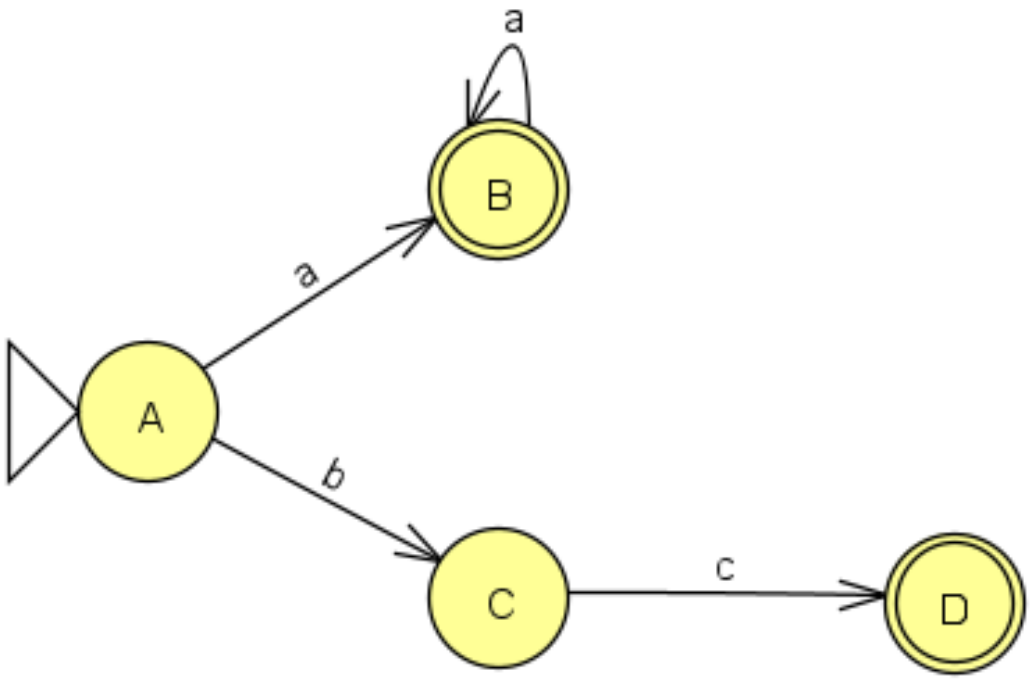
$c - \varepsilon((mueve(C, c)) = c - \varepsilon(7) = \{7, 8\}$ $D = \{7, 8\}$

Nuevo AFD:

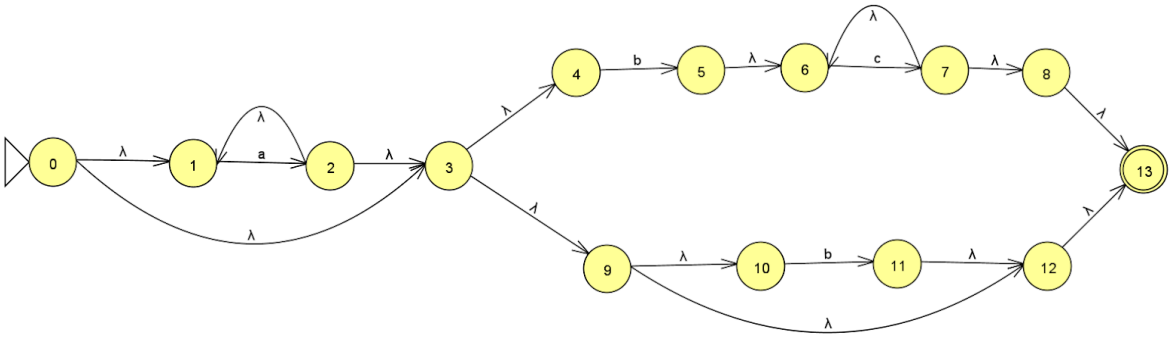
$s = \{A, B, C, D\}$ $s_0 = \{A\}$

$\Sigma = \{a, b, c\}$ $F = \{D, B\}$

Conjunto de estados AFD	a	b	c
$\rightarrow A$	*B*	*C*	-
B_F	*B*	-	-
C	-	-	*D*
D_F	-	-	-



Antiguo AFN:



$s = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\}$ $s_0 = \{0\}$

$\Sigma = \{a, b, c\}$ $F = \{13\}$

$RegExp = a * (\ b\,(c\ +)|\ b\,?)$

$c - \varepsilon(0) = \{0, 1, 3, 4, 9, 10, 12, 13\}$
$A = \{0, 1, 3, 4, 9, 10, 12, 13\}$

$c - \varepsilon((mueve(A, a)) = c - \varepsilon(2) = \{2, 1, 3, 4, 9, 10, 12, 13\}$
$B = \{1, 2, 3, 4, 9, 10, 12, 13\}$

$c - \varepsilon((mueve(A, b)) = c - \varepsilon(5, 11) = \{5, 11, 6, 12, 13\}$
$C = \{5, 6, 11, 12, 13\}$

$c - \varepsilon((mueve(B, a)) = c - \varepsilon(2) = \{2, 1, 3, 4, 9, 10, 12, 13\}$
$B = \{1, 2, 3, 4, 9, 10, 12, 13\}$

$c - \varepsilon((mueve(B, b)) = c - \varepsilon(5, 11) = \{5, 11, 6, 12, 13\}$
$C = \{5, 6, 11, 12, 13\}$

$c - \varepsilon((mueve(C, c)) = c - \varepsilon(7) = \{7, 6, 8, 13\}$
$D = \{6, 7, 8, 13\}$

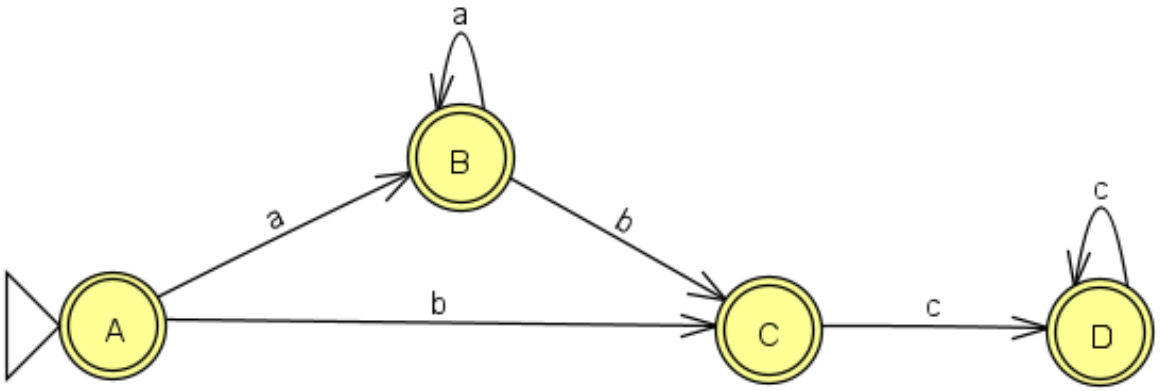
$c - \varepsilon((mueve(D, c)) = c - \varepsilon(7) = \{7, 6, 8, 13\}$
$D = \{6, 7, 8, 13\}$

Nuevo AFD:

$s = \{A, B, C, D\}$ $\quad s_0 = \{A\}$

$\Sigma = \{a, b, c\}$ $\quad F = \{A, B, C, D\}$

Conjunto de estados AFD	a	b	c
$\rightarrow A_F$	*B*	*C*	-
B_F	*B*	*C*	-
C_F	-	-	*D*
D_F	-	-	*D*



Conclusión:

La finalización del programa de implementación del Algoritmo de Conjuntos supone un hito importante en la automatización de procesos que dependen de la manipulación de lenguajes formales. Se ha conseguido una mayor accesibilidad y usabilidad ampliando la interfaz gráfica de usuario existente para que el usuario pueda seleccionar el tema de la Construcción de Conjuntos, facilitando su aplicación en diversos contextos.

El desarrollo y la programación de esta interfaz gráfica demuestran la importancia de la interacción entre la teoría computacional y la informática práctica. Para generar una herramienta útil y amigable para los usuarios, los programadores han tenido que combinar conocimientos en teoría de autómatas con habilidades de diseño de software.

El éxito de las pruebas del programa demuestra la eficacia con la que se ha implementado el Algoritmo Conjunto. Estas pruebas han confirmado que el programa puede traducir con precisión Autómatas Finitos No Deterministas (AFN) a Autómatas Finitos Deterministas (AFD) y producir la correspondiente tabla de transposiciones. En aplicaciones realistas, esta habilidad es esencial para asegurar la exactitud y confiabilidad de la herramienta.

En conclusión, el éxito de las pruebas y la finalización del programa de implementación del Algoritmo de Conjuntos reflejan un importante paso adelante en el desarrollo del futuro Analizador Léxico.