

TECNOLÓGICO NACIONAL DE MEXICO

INSTITUTO TECNOLÓGICO DE IZTAPALAPA

NOMBRE: FERMIN CRUZ ERIK

MATRICULA: 181080007

GRUPO: ISC-6AM

MATERIA: LENGUAJES Y AUTOMATAS I

PROFESOR: M.C. ABIEL TOMÁS PARRA HERNÁNDEZ

ACTIVIDADES SEMANALES

SEMANA 1

TEST DE PERSONALIDAD

Mi personalidad es que soy una persona fuerte, seria, tranquila, un poco sociable, leal, bastante observadora y apasionada por lo que le gusta.

El test de personalidad dice que soy un ser con tipo de personalidad Cónsul, en cuanto a esto se refiere, sí, soy un tanto extrovertido ya que me gusta estar en constante movimiento, de lo contrario suelo aburrirme rápido y por lo mismo, siempre busco algo que hacer; soy bastante observador, también, me gusta analizar bien la situación en la cual me encuentre y tomar la decisión que más me convenga en ese momento; en cuanto a la naturaleza dice que soy emocional y ahí difiero un poco porque suelo ser más una persona de pensamientos y lógica que de emociones.

SEMANA 2

CRÍTICAS SOBRE LOS 4 VÍDEOS (LENGUAJES Y AUTÓMATAS)

VIDEO PARO DE 1 DIA SIN MUJERES

En el primer video en el que hablaron sobre el paro que hicieron las mujeres por 1 día creo que es interesante de hablar no solo que sea un paro digamos a nivel nacional de mujeres en el ámbito laboral y en todos los demás, sino que quizá hubiese pasado lo mismo si hubiese sido al revés (de hombres) porque finalmente alguien tiene que hacer ese trabajo, sea él quien realice o no.

En cuanto al tema sobre el aborto, este siempre ha sido un tema de sensibilidad y debate por mucho tiempo, en lo personal me parece un arma de doble filo porque este podría ser legal pero solamente para aquellas mujeres que hayan sido violentadas como en muchas ocasiones en las que ha pasado, pero, por otro lado, hacerlo legal podría ser contraproducente ya que estarían fomentando el hecho de que si no quieren tenerlo simple y sencillamente lo abortan y ya, es decidir “tomar una vida por otra”.

VIDEO 2. CONVERSANDO CON CRISTINA PACHECO – JULIETA FIERRO

Esta plática entre estos 2 personajes es bueno, por una parte por la profesora en astronomía que ha tenido bastantes horas de estudio no solo para ejercer como profesional sino por el hecho de que le gusta hacerlo y compartir lo que ha aprendido con el paso del estudio y del tiempo y más por la razón de aprender, no se queda con lo que sabe cuándo lo sabe sino que va más allá de lo normal y lo básico, continúa con lo suyo hasta descubrir algo nuevo y seguir así en ese camino; lo mismo que ocurre con la conductora, tener que aprender sobre temas que quizá no domina y que tal vez tampoco ha visto o solo una pequeña parte pero el esforzarse por saber de qué va a hablar su invitado(a) es gratificante, es algo que le va a quedar marcado toda la vida y que se ve que disfruta hacer.

No se queda solo con una pequeña idea de saber de qué le van a hablar en el transcurso del programa, sino que hace una pequeña investigación para tener de esa manera algo que decir y sobre qué platicar y no quedarse solo como estaba hasta antes de la entrevista y eso es algo bueno porque se queda con ese conocimiento para sí misma y después transmitirlo.

VIDEO 3. NOAM CHOMSKY

Con respecto a este personaje bastante importante a decir verdad, me parece que tiene razón en cuanto a lo que se refiere al poder que se les ha dado a los falsos “intelectuales” que ciertamente se les considera en ese término por las responsabilidades que otros ponen en ellos por el lugar en el que se les ha puesto, citando con esto lo que dice Chomsky: “los que son votados, no son los elegidos”, sino que están los que se ponen y que quien decide eso es simplemente una minoría, la cual tiene el poder para hacerlo.

Y que con este mismo, ha logrado hacer bastante para bien o para mal, como hemos visto a lo largo de la historia, los intereses de la minoría, los intereses por los que se ha hecho lucha o revolución han sido justamente de estos mismos personajes, estos grupos que probablemente son minoría pero que han logrado llevar las riendas de la población o inclusive del mundo mismo bajo su yugo, caso de ejemplo me sirva decir el movimiento de independencia en México, el descontento era principalmente, - por no decir solo de éstos- de los criollos que, resentidos con sus padres y antecesores, deseaban el poder que se les había quitado y que creían merecer por derecho.

Logrando hacer que el pueblo se uniera a su causa creyendo que serían libres y que tendrían mejores oportunidades en cualquier aspecto, que serían reconocidos y que todo iba a estar mejor de cómo estaban, fue bueno, sí, pero no deja de ser manipulación para ponerlos bajo sus intereses.

VIDEO 4. ALFREDO JALIFE EN ENTREVISTA CON JULIO ASTILLERO 29.03.19

En este último video me agrada la idea de atacar las primicias del gabinete presidencial, el hablar de la política del país que, si recordamos, son aquellos que se suponían, iba a encerrar, cosa que no paso y que incluso fue caso contrario: se les dio “el perdón” a más de la mitad por no decir que a todos esos personajes de antaño que tanto daño y mal hicieron al país, es interesante ver como se desenvuelve y va de un tema a otro pero sin perder el hilo, eso es bueno y gratificante, hace la plática amena a pesar de tocar temas un tanto sensibles y que, en lo personal no tengo bastante información o no sé de todo sobre lo que habla pero al escucharlo uno se da una idea de qué es lo que quiere dar a entender con lo que pronuncia.

Me agrada la sátira con la que ataca todo este tema de la 4T y la nueva administración porque... es cierto, pero a la vez también es triste y risible, una sensación que probablemente tendría cualquier persona ante una situación así (sabiendo que tiene un veto por un reto que hizo el sexenio pasado con Felipe Calderón). Es una persona que en general, sabe bastante del tema y sabe cómo compartirlo.

SEMANA 3

Buenas tardes profesor, anexo las ligas de las cuentas para la plataforma de github

Materia: LENGUAJES Y AUTOMATAS I

Grupo: ISC-6AM

Integrantes

Ligas de las cuentas

Cuananemi Cuanalo Mario Alberto: <https://github.com/cuanenemi94>

Fermin Cruz Erik: <https://github.com/erikfcz>

Gutierrez Arellano Rafael: <https://github.com/Rafa6000>

Perez Armas Fausto Isaac: <https://github.com/FaustoArmas>

SEMANA 4

En la introducción tocaremos de primera mano una leve información acerca de lo que veremos en esta materia, los temas que iremos viendo poco a poco conforme avancemos mediante el transcurso del tiempo, primeramente comenzaremos con el tema de “Los lenguajes y su representación finita”.

Primeramente comenzamos a ver algunos temas del curso, por ejemplo la computación la cual se emplea para resolver algunos de los problemas a través de la mecánica, haciendo uso de programaciones en estas mismas y en pasos inequívocos, con la finalidad de ayudar a resolver cuestiones que quizá serían más difíciles sin las computadoras.

Seguidamente de esto, nos mencionan que la computación puede ser realizada por una computadora (valga ahora la redundancia), y finalmente, vemos que la teoría computación comprende la propiedad matemática fundamental de una computadora, hardware y software.

Para hacer uso de la computación necesitamos tener en cuenta que requerimos de algoritmos y estructuras de código para poder mandar a ejecutar una instrucción, sin estas, no es posible realizar el trabajo que deseamos realizar.

En el curso vienen diversos temas, algunos de los cuales son: alfabeto, cadena, lenguajes, tipos de lenguajes, por decir solo algunos de ellos, por ejemplo.

En la teoría de la computación explican sobre qué es la teoría de la computación, qué es lo que puede hacer, cuáles son sus limitaciones y cuál es la complejidad de una computadora. dentro de este mismo vemos que existen lenguajes para poder resolver estos problemas, sin embargo, como bien se menciona en el video, dependerá también del nivel de dificultad al cual nos enfrentemos.

Finalmente tendremos lo que son los tipos de modelos y su clasificación, por ejemplo la máquina de turing, reconocida por haber sido aquella máquina capaz de descifrar los mensajes encriptados de los alemanes, y con la cual, se ganaría básicamente la guerra, poniendo fin a la misma.

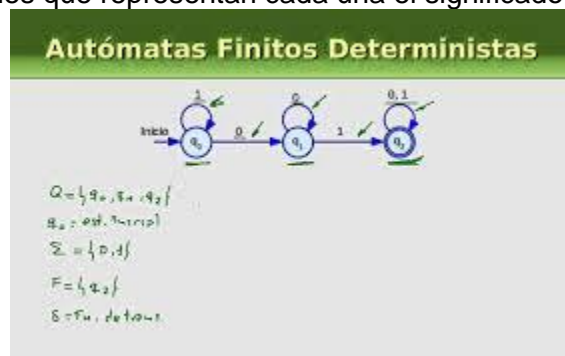
SEMANA 5

bueno en estos videos siguientes nos dan la continuidad de los que ya habíamos visto y que nos hablaban sobre lo que son los autómatas y la gramática pero solamente como una breve introducción, en este caso ya vamos a verlos pero con más profundidad.

bueno primeramente, tenemos que la concatenación es asociativa, aquí nos dice que si tenemos por ejemplo 3 lenguajes, L1, L2 y L3, entonces la concatenación de L1 es equivalente a L2 y L2 es equivalente a la concatenación de L3, aunque igualmente hay caso en los que L1 puede no ser igual a L2, teniendo en consecuencia una no asociación de lenguajes debido a que ambas serian diferentes cadenas.

L1L2 diferente de L2L1

como tal un autómata nos lo tienen definido como un conjunto de estados los cuales están definidos por un número finito de símbolos de entrada o mejor conocido como alfabeto, donde tenemos variables que representan cada una el significado de esto.



en la representación finita tenemos que es un número determinado o finito de pasos a seguir para resolver el problema dado, es decir, un algoritmo con las instrucciones necesarias para resolver la situación o problema dado.

En esta parte entendemos que un autómata finito o como también es conocido "máquina de estado finito" se refiere a un modelo computacional que realiza operaciones en forma automática sobre una entrada para producir una salida, es decir, obtención de un resultado a partir de los datos ingresados o pedidos.

Este modelo está conformado como bien nos lo mencionan en los videos pasados y recientes, por un alfabeto, un conjunto de estados finitos, una función de transición, un estado inicial y un conjunto de estados finales.

Su funcionamiento se basa principalmente en una función de transición, que recibe a partir de un *estado inicial* una cadena de caracteres pertenecientes al alfabeto (lo que es la entrada), y que va leyendo dicha cadena a medida que el autómata se desplaza de un

estado a otro, para finalmente detenerse en un *estado final* o *de aceptación*, que representa la salida.

La finalidad de los autómatas finitos es reconocer lenguajes regulares, que corresponden a los lenguajes formales más simples según la Jerarquía de Chomsky.

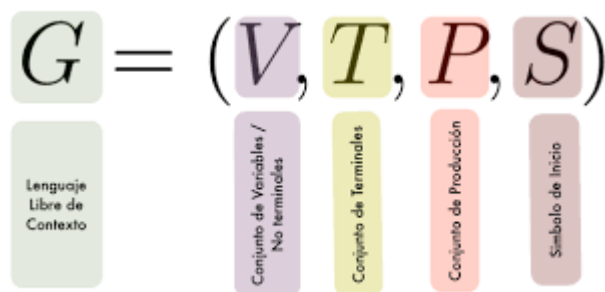
Gramática

La gramática en los autómatas se refiere básicamente a las normas que describen la secuencia de símbolos de un conjunto. Esta tiene cuatro elementos fundamentales:

$$G = \{ NT, T, S, P \}$$

En donde:

- NT es el conjunto de elementos **No Terminales**,
- T es el conjunto de elementos **Terminales**,
- S es el **Símbolo inicial** de la gramática y finalmente,
- P es el conjunto de **Reglas de Producción**



en esta imagen podemos ver los elementos anteriormente definidos.

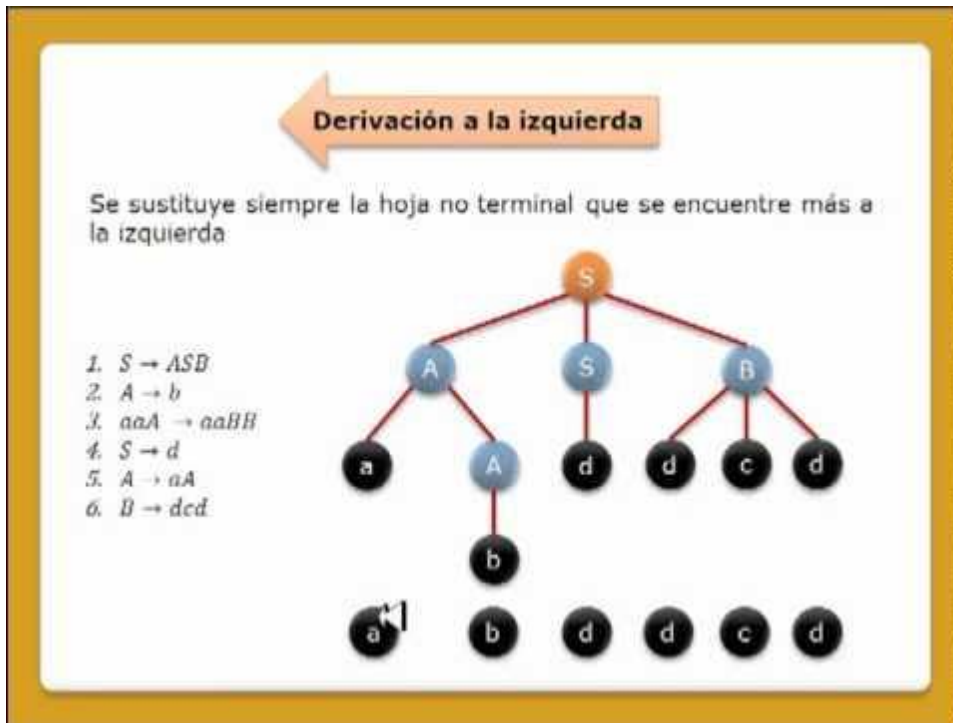
La gramática se puede clasificar según Padilla en 4 tipos:

- 1.- No restringida o recursivamente enumerables
- 2.- Sensible al contexto"
- 3.- libre de contexto"
- 4.- De contexto regular"

ÁRBOLES DE DERIVACIÓN

Finalmente tenemos el último tema que es el de los árboles de derivación. Estos nos dicen que son aquellos que nos permiten mostrar gráficamente cómo podemos derivar cualquier cadena de un lenguaje a partir del símbolo distinguido de una gramática que genera ese lenguaje.

En su construcción tenemos que cada nodo del árbol va a contener un símbolo. En el nodo raíz se pone el símbolo inicial S y se efectúa una ramificación del árbol por cada producción que se aplique.



SEMANA 6

GRAMÁTICA REGULAR

La gramática regular en lo autómatas se refiere a una gramática que puede ser clasificada de 2 maneras: de izquierda o de derecha. Estas gramáticas pueden dar paso a los lenguajes que de igual manera llevan su nombre: lenguajes regulares.

se utilizan principalmente para analizar el contenido que tenemos en una cadena de caracteres por medio de patrones, estos patrones son usados para identificar una determinada combinación de estos caracteres dentro de un string o cadena de tipo texto.

En la gramática regular de derecha tenemos que A entonces a . si es un símbolo que no termina en N y termina en Σ .

Mientras que en la gramática de izquierda tenemos algunas reglas, que son:

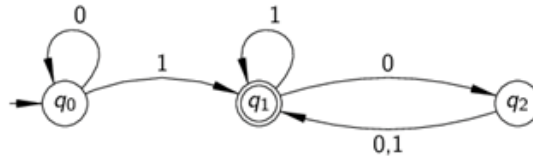
- $A \rightarrow a$, donde A es un símbolo no-terminal en N y a uno terminal en Σ
- $A \rightarrow Ba$, donde A y B pertenecen a N y a pertenece a Σ
- $A \rightarrow \epsilon$, donde A pertenece a N .

AUTOMATA FINITO

Se trata de un modelo computacional que realiza cálculos o instrucciones de manera automática en una entrada para producir una salida.

Este modelo está conformado por varias partes como por ejemplo un alfabeto, un conjunto de estados finitos, una función de transiciones, un estado inicial y por último, un estado final.

Recordad los autómatas finitos deterministas



- ▶ La **computación** del autómata con entrada 011 es (q_0, q_0, q_1, q_1) que me dice la secuencia de estados por los que pasa con entrada 011
- ▶ Cada entrada me da exactamente una computación. Tengo siempre como mucho **una opción** desde un estado si leo un símbolo ← Esto se llama **determinismo**

Su funcionamiento se basa en la ausencia de redundancia, en una función de transición, que recibe a partir de un estado inicial una cadena de caracteres pertenecientes al alfabeto (la entrada), y que va leyendo dicha cadena a medida que el autómata se desplaza de un estado a otro, para finalmente detenerse en un estado final o de aceptación, que representa la salida.

La finalidad de los autómatas finitos es la de reconocer lenguajes regulares, que corresponden a los lenguajes formales más simples según la Jerarquía de Chomsky.

Estos tienen su origen en las máquinas electromecánicas, en las cuales el matemático ruso Andréi Márkov formalizó un proceso llamado cadena de Markov, donde la ocurrencia de cada evento dependía de una cierta probabilidad del evento anterior. Esta particularidad de recordar los estados anteriores, les sirve de mucho a los autómatas para poder tomar el siguiente estado, ya que deben conocer el anterior para saber cuál es el posterior.

AUTÓMATA FINITO NO DETERMINISTA

Son autómatas que, a diferencia de los autómatas finitos deterministas, poseen al menos un estado $q \in Q$, tal que para un símbolo $a \in \Sigma$ del alfabeto, existe más de una transición posible.

Estos pueden darse en cualquiera de estos dos casos:

- 1.- Que existan transiciones del tipo $\delta(q,a)=q_1$ y $\delta(q,a)=q_2$, siendo $q_1 \neq q_2$;

2.- Que existan transiciones del tipo $\delta(q, \epsilon)$, siendo q un estado no-final, o bien un estado final pero con transiciones hacia otros estados.

Los autómatas no deterministas pueden pasar por varios estados a la vez, generando una ramificación de las configuraciones existentes en un momento dado. Asimismo, en estos podemos aceptar la existencia de más de un nodo inicial.

Estos autómatas tienen un funcionamiento que es por decirlo así, de la siguiente manera:

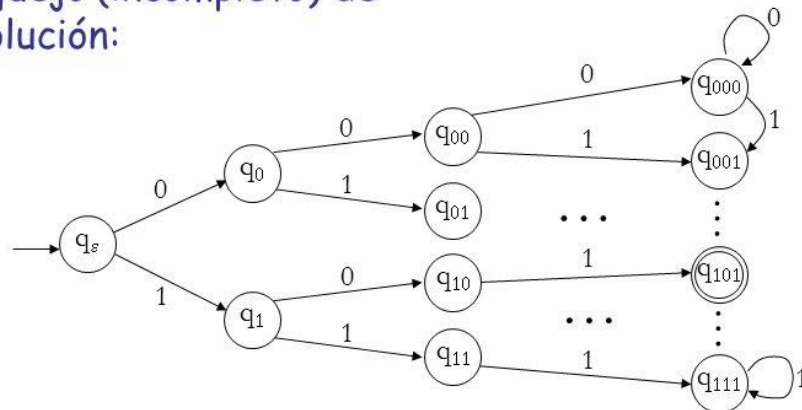
La máquina comienza en el estado inicial que se le ha especificado y lee una cadena de caracteres pertenecientes al alfabeto. El autómata utiliza la función de transición de estados T para determinar el siguiente estado, usando el estado actual y el símbolo que acaba de leer o la cadena vacía. Sin embargo, "el estado siguiente de un Autómata finito no determinado no solo depende del evento de la entrada actual, sino que también depende de un número arbitrario de los eventos de entrada posterior.

Hasta que se producen estos acontecimientos posteriores no es posible determinar en qué estado se encuentra la máquina". Cuando el autómata ha terminado de leer, y pasa a un estado de aceptación, se dice que acepta la cadena, de lo contrario se dice que la cadena de caracteres es rechazada.

AFND: Autómatas finitos *no deterministas*

Veamos un ejemplo más de AFD. Con $\Sigma=\{0,1\}$, queremos que acepte el lenguaje de los strings que terminan en 101.

Bosquejo (incompleto) de la solución:



SEMANA 7

DFA: Se refiere a un autómata finito determinista. Se dice que un autómata finito es determinista, si corresponde a un símbolo de entrada, y en este hay un solo estado resultante, es decir, solo hay una transición.

Un autómata finito determinista está formado por cinco tuplas y se representa como:

Donde:

Q: Un conjunto finito no vacío de estados presentes en el control finito (q_0, q_1, q_2, \dots).

Σ : Un conjunto finito no vacío de símbolos de entrada.

δ : Es una función de transición que toma dos argumentos, un estado y un símbolo de entrada, devuelve un solo estado.

q_0 : Es el estado inicial, uno de los estados en Q.

F: Es un conjunto no vacío de estados finales / estados de aceptación del conjunto que pertenece a Q.

NFA: Se refiere a un autómata finito no determinista. Se dice que un autómata finito no es determinista, si hay más de una posible transición desde un estado en el mismo símbolo de entrada.

Un autómata finito no determinista también está formado por cinco tuplas y se representa de la siguiente manera:

Donde:

Q: Un conjunto de estados finitos no vacíos.

Σ : Un conjunto de símbolos de entrada finitos no vacíos.

δ : Es una función de transición que toma un estado de Q y un símbolo de entrada y devuelve un subconjunto de Q.

q_0 : Estado inicial de NFA y miembro de Q.

F: Un conjunto no vacío de estados finales y miembro de Q.

TEOREMA DE NERODE

Caracteriza los lenguajes regulares como subconjuntos del monoide que son saturados por una congruencia de índice infinito.

Sea 'S' un semigrupo. Un subconjunto X de S se llama reconocible si está saturado por una congruencia en 'S' de índice finito. Denotamos por Rec (S) el conjunto formado por todos los subconjuntos de 'S' reconocibles. Sea 'S' un semigrupo y 'X' un subconjunto de 'S'. Se define la congruencia sintáctica de X como $s \approx_X t \Leftrightarrow \forall u, v \in S (usv \in X \Leftrightarrow utv \in X)$

MINIMIZACIÓN DE UNA NFA

Si bien una búsqueda exhaustiva puede minimizar un NFA, no existe un algoritmo de tiempo polinomial para minimizar los NFA generales a menos que $P = PSPACE$, una conjetura no resuelta en la teoría de la complejidad computacional que se cree ampliamente que es falsa.

Sin embargo, existen métodos de minimización de NFA que pueden ser más eficientes que la búsqueda por fuerza bruta.

SEMANA 8

EXPRESIONES REGULARES

Es un equivalente algebraico para un autómata utilizado en muchos lugares como un lenguaje para describir patrones en texto que son sencillos pero muy útiles.

Pueden definir exactamente los mismos lenguajes que los autómatas pueden describir: Lenguajes regulares

Ofrecen algo que los autómatas no: Manera declarativa de expresar las cadenas que queremos aceptar.

Ejemplos de sus usos:

- Comandos de búsqueda, e.g., grep de UNIX
- Sistemas de formateo de texto: Usan notación de tipo expresión regular para describir patrones
- Convierte la expresión regular a un DFA o un NFA y simula el autómata en el archivo de búsqueda.
- Generadores de analizadores-léxicos. Como Lex o Flex.
- Los analizadores léxicos son parte de un compilador. Dividen el programa fuente en unidades lógicas(tokens), como while, números, signos (+, -, <, etc.)
- Produce un DFA que reconoce el token.

OPERANDOS

Si E y F son expresiones regulares, entonces $E + F$ también lo es denotando la unión de $L(E)$ y $L(F)$. $L(E + F) = L(E) \cup L(F)$.

Si E y F son expresiones regulares, entonces EF también lo es denotando la concatenación de $L(E)$ y $L(F)$. $L(EF) = L(E)L(F)$.

EJEMPLO PARA TRANSFORMAR UN DFA EN UNA EXPRESIÓN REGULAR

Ahora, vamos a ver uno de los métodos que se usan para transformar autómatas finitos deterministas en expresiones regulares, el método de eliminación de estados. Cuando tenemos un autómata finito, determinista o no determinista, podemos considerar que los símbolos que componen a sus transiciones son expresiones regulares. Cuando eliminamos un estado, tenemos que reemplazar todos los caminos que pasaban a través de él como transiciones directas que ahora se realizan con el ingreso de expresiones regulares, en vez de con símbolos. Los casos bases son los siguientes:

Realmente, el retorno podría verse como un caso particular de la unión, en donde q_0 y q_1 son el mismo estado. De esta forma, el camino que va directo desde q_0 a q_0 es "Y" y el que va desde q_0 a q_0 a través de q_2 es "VW*X".

A la hora de reducir un autómata, se recomienda partir eliminando primero todos los estados que no sean ni el de inicial ni los finales. Cuando se eliminan todos estos estados y el autómata tenga más de un estado inicial, se deben hacer tantas copias como estados de aceptación tenga el autómata.

En cada una de las copias, se debe elegir uno de los estados de aceptación diferentes. Todos los demás estados de aceptación de esta copia pasarán a ser estados ordinarios. Ahora se deben reducir todos los autómatas copias a expresiones regulares. La expresión regular final será la unión de todas las expresiones regulares resultantes de cada una de las copias.

SEMANA 10

EXPRESIONES REGULARES

Es un equivalente algebraico para un autómata utilizado en muchos lugares como un lenguaje para describir patrones en texto que son sencillos pero muy útiles.

Pueden definir exactamente los mismos lenguajes que los autómatas pueden describir:

Lenguajes regulares Ofrecen algo que los autómatas no: Manera declarativa de expresar las cadenas que queremos aceptar.

Ejemplos de sus usos:

- Comandos de búsqueda, e.g., grep de UNIX
- Sistemas de formateo de texto: Usan notación de tipo expresión regular para describir patrones
- Convierte la expresión regular a un DFA o un NFA y simula el autómata en el archivo de búsqueda.
- Generadores de analizadores-léxicos. Como Lex o Flex.
- Los analizadores léxicos son parte de un compilador. Dividen el programa fuente en unidades lógicas(tokens), como while, números, signos (+, -, <, etc.)
- Produce un DFA que reconoce el token.

OPERANDOS Si E y F son expresiones regulares, entonces $E+F$ también lo es denotando la unión de $L(E)$ y $L(F)$. $L(E+F) = L(E) \cup L(F)$. Si E y F son expresiones regulares, entonces EF también lo es denotando la concatenación de $L(E)$ y $L(F)$. $L(EF) = L(E)L(F)$.

EJEMPLO PARA TRANSFORMAR UN DFA EN UNA EXPRESIÓN REGULAR

Ahora, vamos a ver uno de los métodos que se usan para transformar autómatas finitos deterministas en expresiones regulares, el método de eliminación de estados. Cuando tenemos un autómata finito, determinista o no determinista, podemos considerar que los símbolos que componen a sus transiciones son expresiones regulares. Cuando eliminamos un estado, tenemos que reemplazar todos los caminos que pasaban através de él como transiciones directas que ahora se realizan con el ingreso de expresiones regulares, en vez de con símbolos. Los casos bases son los siguientes:

Realmente, el retorno podría verse como un caso particular de la unión, en donde q_0 y q_1 son el mismo estado. De esta forma, el camino que va directo desde q_0 a q_0 es "Y" y el que va desde q_0 a q_0 a través de q_2 es "VW*X".

A la hora de reducir un autómata, se recomienda partir eliminando primero todos los estados que no sean ni el de inicial ni los finales. Cuando se eliminen todos estos estados y el autómata tenga más de un estado inicial, se deben hacer tantas copias como estados de aceptación tenga el autómata.

En cada una de las copias, se debe elegir uno de los estados de aceptación diferentes. Todos los demás estados de aceptación de esta copia pasarán a ser estados ordinarios. Ahora se deben reducir todos los autómatas copias a expresiones regulares. La expresión regular final será la unión de todas las expresiones regulares resultantes de cada una de las copias.




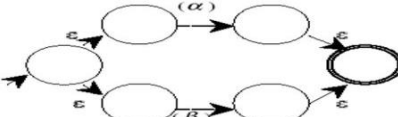
SEMANA 11

EXPRESIÓN REGULAR

Los lenguajes aceptados por un AF son fácilmente descritos por una expresión llamada Expresión regular. Esto quiere decir que:

Sea E un conjunto finito de símbolos y sean L, L1 y L2 conjunto de cadenas de E, la concatenación de L1 y L2, denotada por L_1L_2 , es el conjunto $\{xy\}$ donde x esta en L1 e Y esta e L2}.

EJERCICIOS DE EXPRESIONES REGULARES Y AUTOMATAS

Expresión Regular	Autómata finito
\emptyset	
ϵ	
$a \in \Sigma$	
$\alpha \mid \beta$	

EJEMPLOS:

Sea $E = \{0, 1\}$ y sea $R = 0^*1+0$ y queremos construir un autómata cuyo lenguaje sea exactamente el definido por la expresión regular r . El último operador que interviene es la suma.

SEMANA 12

AUTÓMATAS FINITOS Y EXPRESIONES REGULARES

Un autómata finito como tal se refiere a un modelo computacional que realiza cálculos en forma automática sobre una entrada para producir una salida.

Este modelo está conformado por un alfabeto, un conjunto de estados finitos, una función de transición, un estado inicial y un conjunto de estados finales.

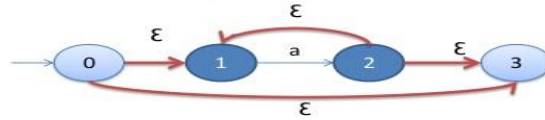
Su funcionamiento se basa en una función de transición, que recibe a partir de un estado inicial una cadena de caracteres pertenecientes al alfabeto (la entrada), y que va leyendo dicha cadena a medida que el autómata se desplaza de un estado a otro, para finalmente detenerse en un estado final o de aceptación, que representa la salida.

EXPRESIONES REGULARES EN AUTÓMATAS FINITOS

Los lenguajes descritos por expresiones regulares son los lenguajes reconocidos por los autómatas finitos. Existe un algoritmo para convertir una expresión regular en el autómata finito no determinístico correspondiente. El algoritmo construye a partir de la expresión regular un autómata con transiciones vacías, es decir un autómata que contiene arcos rotulados con ϵ . Luego este autómata con transiciones vacías se puede convertir en un autómata finito sin transiciones vacías que reconoce el mismo lenguaje.

- Dada una expresión regular existe un autómata finito capaz de reconocer el lenguaje que ésta define.
- Recíprocamente, dado un autómata finito, se puede expresar mediante una expresión regular del lenguaje que reconoce.

Ejemplo



Cerradura ϵ de un estado: El estado mismo y los estados que conduce una transición ϵ

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

$$\bar{1} = \{1\}$$

$$\bar{2} = \{1, 2, 3\}$$

$$\bar{3} = \{3\}$$

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

SEMANA 13

Al igual que las expresiones aritméticas, las expresiones regulares cumplen una serie de leyes. Muchas de éstas son similares a las leyes aritméticas, si interpretamos la unión como una suma y la concatenación como una multiplicación. También existen algunas leyes que se aplican a las expresiones regulares, pero no tienen su análoga en la aritmética, especialmente cuando se utiliza el operador de clausura.

Asociatividad y conmutatividad

La conmutatividad es la propiedad de un operador que establece que se puede cambiar el orden de sus operandos y obtener el mismo resultado. Anteriormente hemos dado un ejemplo para la aritmética: $x+y=y+x$.

La asociatividad es la propiedad de un operador que nos permite reagrupar los operandos cuando el operador se aplica dos veces. Por ejemplo, la ley asociativa de la multiplicación es $(xxy) \times z = xx(yxz)$.

Las tres leyes de este tipo que se cumplen para las expresiones regulares son:

- $L+M = M+L$. Esta ley, la ley conmutativa de la unión, establece que podemos efectuar la unión de dos lenguajes en cualquier orden.
- $(L+M)+N = L+(M+N)$. Esta ley, la ley asociativa para la unión, establece que podemos efectuar la unión de tres lenguajes bien calculando primero la unión de los dos primeros, o bien la unión de los dos últimos. Obsérvese que, junto con la ley conmutativa de la unión, podemos concluir que es posible obtener la unión de cualquier colección de lenguajes en cualquier orden y agrupamiento, y el resultado siempre será el mismo. Intuitivamente, una cadena pertenece a $L_1 \cup L_2 \cup \dots \cup L_k$ si y sólo si pertenece a uno o más de los L_i

- $(LM)N = L(MN)$. Esta ley, la ley asociativa para la concatenación, establece que podemos concatenar tres lenguajes concatenando primero los dos primeros o bien los dos últimos.

Falta en esta lista la “ley” que establece que $LM = ML$, es decir, que la concatenación es conmutativa. Sin embargo, esta ley es falsa.

Elemento identidad y elemento nulo

- El elemento identidad de un operador es un valor tal que cuando el operador se aplica al propio elemento identidad y a algún otro valor, el resultado es ese otro valor. Por ejemplo, 0 es el elemento identidad para la suma, ya que $0+x = x+0 = x$, y 1 es el elemento identidad de la multiplicación, puesto que $1 \times x = x \times 1 = x$.
- El elemento nulo de un operador es un valor tal que cuando el operador se aplica al propio elemento nulo y a algún otro valor, el resultado es el elemento nulo. Por ejemplo, 0 es el elemento nulo de la multiplicación, ya que $0 \times x = x \times 0 = 0$. La suma no tiene elemento nulo.

Existen tres leyes para las expresiones regulares que implican estos conceptos y que enumeramos a continuación:

1. $\emptyset + L = L + \emptyset = L$. Esta ley establece que \emptyset es el elemento identidad para la unión.
2. $\epsilon L = L \epsilon = L$. Esta ley establece que ϵ es el elemento identidad para la concatenación.
3. $\emptyset L = L \emptyset = \emptyset$. Esta ley establece que \emptyset es el elemento nulo de la concatenación.

Estas propiedades son importantes herramientas en las tareas de simplificación. Por ejemplo, si se tiene una unión de varias expresiones, algunas de las cuales están simplificadas, o han sido simplificadas, a \emptyset , entonces los \emptyset pueden eliminarse de la unión. Del mismo modo, si tenemos una concatenación de varias expresiones, algunas de las cuales están simplificadas, o han sido simplificadas a ϵ , podemos eliminar los ϵ de la concatenación.

Por último, si tenemos una concatenación de cualquier número de expresiones, y al menos una de ellas es \emptyset , entonces la concatenación completa puede ser reemplazada por \emptyset .

Leyes distributivas

Una ley distributiva implica a dos operadores y establece que un operador puede aplicarse por separado a cada argumento del otro operador. El ejemplo más común en aritmética es la ley distributiva de la multiplicación respecto de la suma, es decir, $x \times (y+z) = x \times y + x \times z$. Puesto que la multiplicación es conmutativa, no importa que la multiplicación esté a la izquierda o a la derecha de la suma. Sin embargo, existe una ley análoga para las expresiones regulares, que tenemos que establecer de dos formas, ya que la concatenación no es conmutativa.

Estas leyes son:

- $L(M+N) = LM+LN$. Ésta es la ley distributiva por la izquierda de la concatenación respecto de la unión.
- $(M+N)L = ML+NL$. Ésta es la ley distributiva por la derecha de la concatenación respecto de la unión.

SEMANA 14

AUTÓMATA PILA

Es una extensión del autómata finito no determinista con transiciones- ϵ , el cual constituye una forma de definir los lenguajes regulares. El autómata a pila es fundamentalmente un AFN- ϵ con la adición de una pila.

La pila se puede leer, se pueden introducir elementos en ella y extraer sólo el elemento que está en la parte superior de la misma, exactamente igual que la estructura de datos de una "pila".

Existen dos versiones diferentes del autómata a pila: una que acepta introduciendo un estado de aceptación, al igual que el autómata finito, y otra versión que acepta vaciando la pila, independientemente del estado en que se encuentre. Estas dos versiones aceptan sólo lenguajes independientes del contexto; es decir, gramáticas que pueden convertirse en autómatas a pila, y viceversa.

La subclase de autómatas a pila que son deterministas aceptan todos los lenguajes regulares, pero sólo un subconjunto adecuado de los lenguajes independientes del contexto. Puesto que son muy similares a los mecanismos del analizador sintáctico de un compilador típico, es importante fijarse en qué construcciones del lenguaje pueden y no pueden reconocer los autómatas a pila deterministas.

El autómata a pila puede observar el símbolo colocado en la parte superior de la pila y llevar a cabo su transición basándose en el estado actual, el símbolo de entrada y el símbolo que hay en la parte superior de la pila. Alternativamente, puede hacer una transición "espontánea", utilizando ϵ como entrada en lugar de un símbolo de entrada.

En una transición, el autómata a pila:

1. Consume de la entrada el símbolo que se usa en la transición. Si como entrada se utiliza ϵ , entonces no se consume ningún símbolo de entrada.
2. Pasa a un nuevo estado, que puede o no ser el mismo que el estado anterior.
3. Reemplaza el símbolo de la parte superior de la pila por cualquier cadena. La cadena puede ser ϵ , lo que corresponde a una extracción de la pila. Podría ser el mismo símbolo que estaba anteriormente en la cima de la pila; es decir, no se realiza ningún cambio en la pila.

También podría reemplazar el símbolo de la cima de la pila por otro símbolo, lo que cambiaría la cima de la pila pero no añade ni extraer ningún símbolo. Por último, el símbolo de la cima de la pila podría ser reemplazado por dos o más símbolos, lo que (posiblemente) tendría el efecto de cambiar el símbolo de la cima de la pila, añadiendo después uno o más nuevos símbolos a la pila.

SEMANA 15

El propósito de la teoría de los problemas indecidibles es proporcionar una guía a los programadores sobre lo que se puede o no conseguir a través de la programación.

La razón de ello es que mientras que los problemas indecidibles normalmente suelen resultar obvios y habitualmente no se intentan resolver, los problemas intratables se presentan continuamente. Además, a menudo dan lugar a pequeñas modificaciones de los

requisitos o a soluciones heurísticas. Por tanto, el diseñador se enfrenta con frecuencia a tener que decidir si un problema es o no intratable, y qué hacer si lo es.

Una ventaja de la máquina de Turing sobre los programas como representación de lo que se puede calcular es que la máquina de Turing es lo suficientemente simple como para que podamos representar su configuración de manera precisa, utilizando una notación sencilla muy similar a las descripciones instantáneas de un autómata a pila.

Con la notación de la máquina de Turing, demostraremos que ciertos problemas, que aparentemente no están relacionados con la programación, son indecidibles.