Presentación del Curso

Gustavo Garzón Villamizar

Autómatas y lenguajes formales
Escuela de Ingeniería de Sistemas e Informática
Universidad Industrial de Santander

3 de septiembre de 2019

Agenda

Información del profesor

- Nombre: Gustavo Garzón Villamizar
- Correo (contacto y comunicación): gustavo.garzon@saber.uis.edu.co

Horarios de clase y atención

	Lunes	**Martes**	**Miércoles**	**Jueves**	**Viernes**
? - ?					

Aula: LP 254 (Sala Villabona)
Atención a estudiantes: LP 338 (Grupo BIVL^{2}ab)

Agenda

Esquema propuesto para la evaluación del curso

Notas		
Descripción	**Porcentaje**	**Total**
Taller 1	10 %	30 %
Parcial 1	20 %	
Taller 2 (a y b)	10 %	30 %
Parcial 2	20 %	
Taller 3	10 %	40 %
Proyecto Final	10 %	
Parcial 3	20 %	

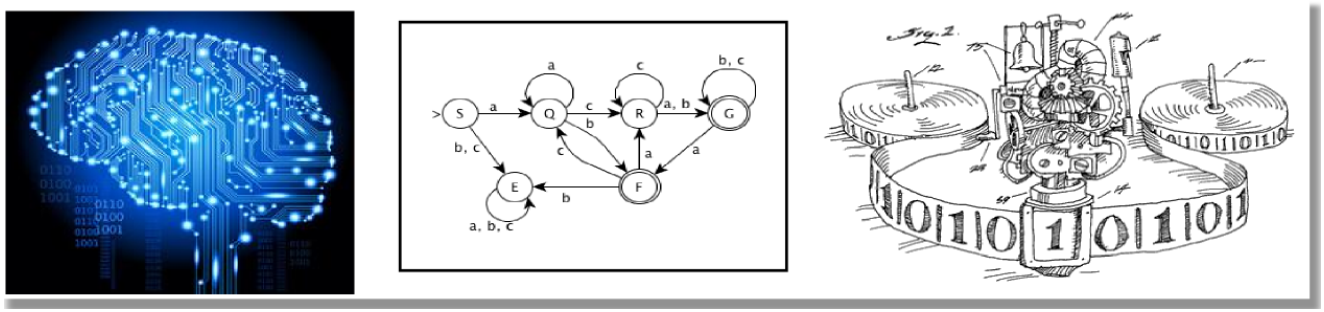
* Los quices que se realicen tendrán efecto si la nota de talleres es menor a 3.0

Primer Parcial

21-25 de Octubre!

Agenda

Definición de Autómatas



Definición

La teoría de autómatas es el estudio de dispositivos de cálculo abstractos (teóricos), es decir, de las “máquinas” que:

- Describen de forma precisa los límites entre lo que una máquina puede y no puede hacer
- Son sistemas que pueden encontrarse siempre en uno de una serie de “estados” finitos

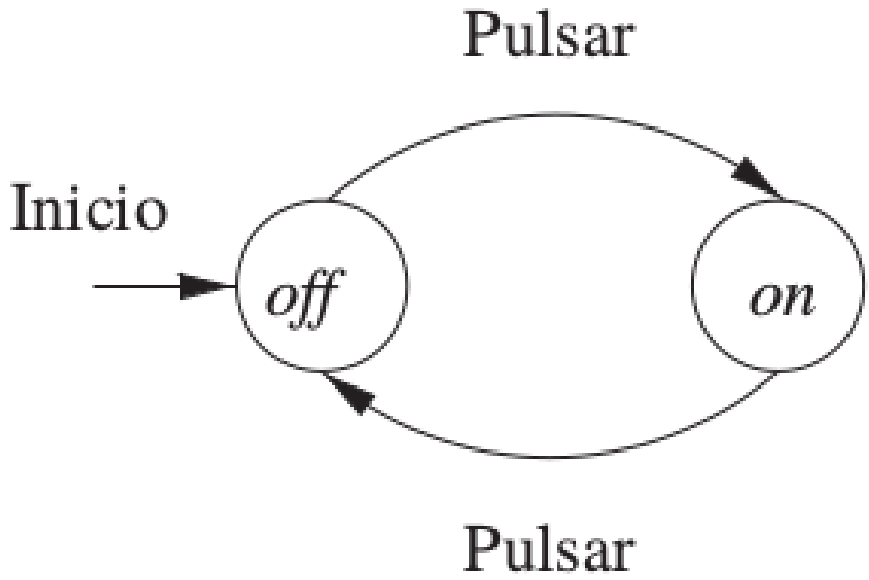
Qué es un Autómata

Dispositivo mecánico o electrónico o biológico

- En un punto de tiempo está en un estado
- Dado una razón (por ejemplo una señal de entrada) cambia de estado

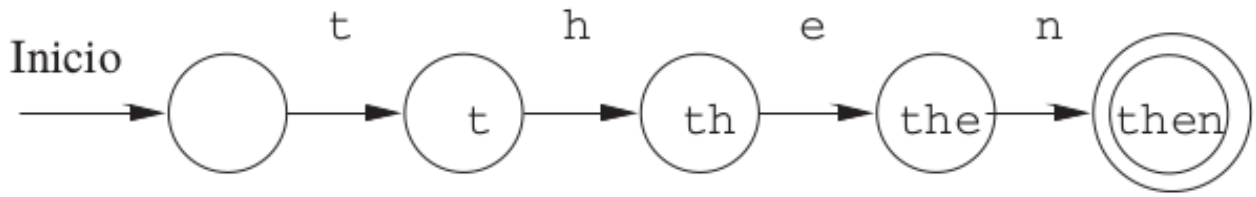
Ejemplos: reloj mecánico o electrónico, máquina para lavar, todo un ordenador

Autómata finito más simple



Interruptor (encendido/apagado *on/off*)

- El dispositivo *recuerda* si el estado es *on* o *off*
- Los estados están representados mediante círculos
- Los arcos son “entradas” externas que influyen en el sistema
- Estado inicial indicado por la palabra *inicio*



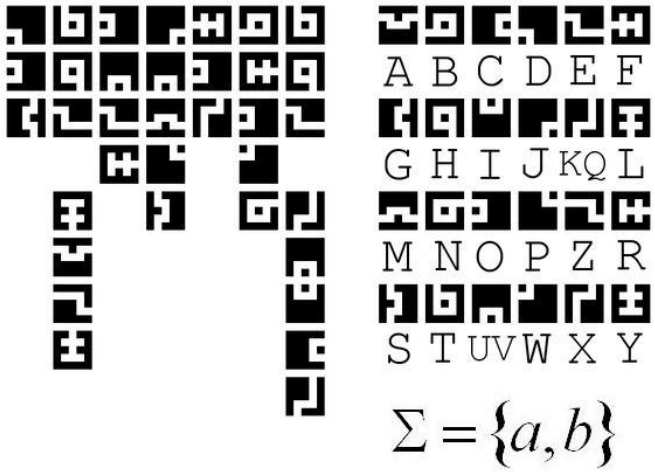
Analizador léxico

Reconoce la palabra clave *then*

- Cada estado corresponde a los prefijos de las palabras
- Estado de aceptación se representa con un círculo doble

Alfabetos y lenguajes

- Introducción a Autómatas Finitos
- Demostraciones formales
- Alfabetos, cadenas de caracteres y lenguajes



$$\Sigma = \{a, b\}$$

$$\Sigma = \{a, b\}$$

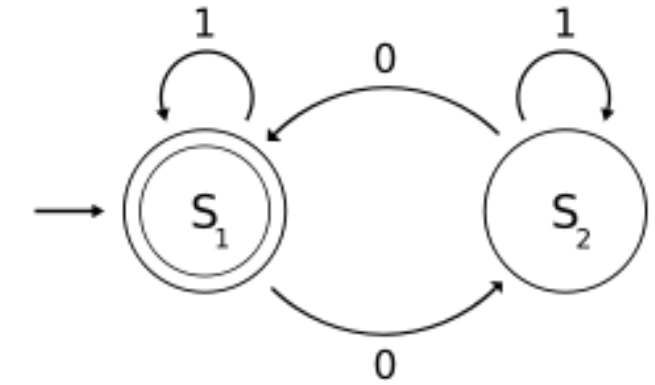
$$u = ab$$

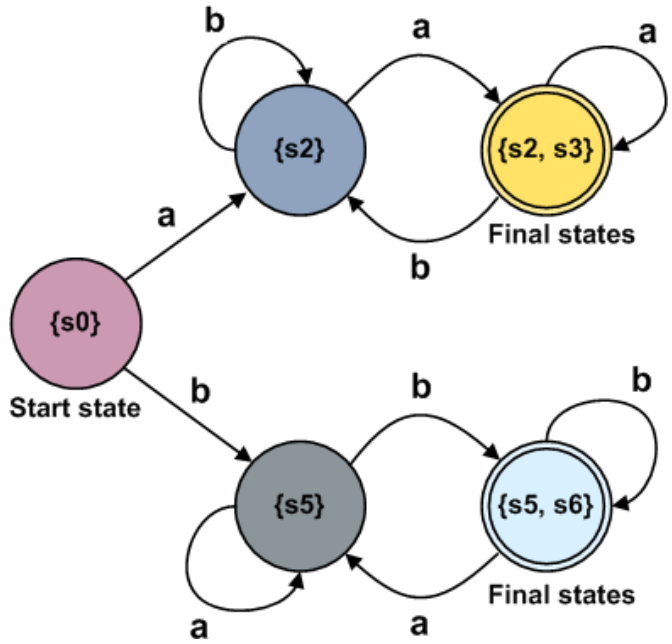
$$v = bbbaaa$$

$$w = abba$$

Autómatas Finitos

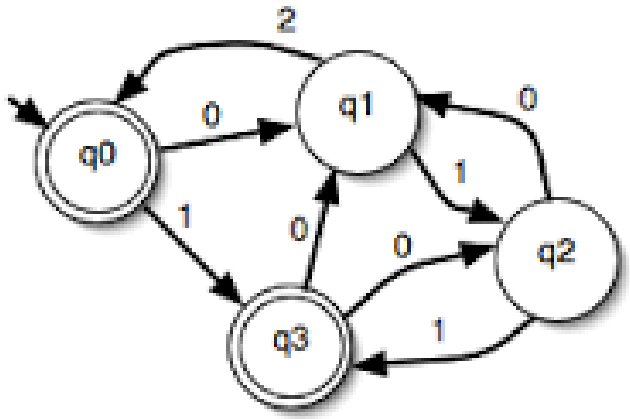
- Definición de autómata
- Autómata finito determinista
- Autómatas finitos no deterministas
- Autómatas finitos con transacciones.

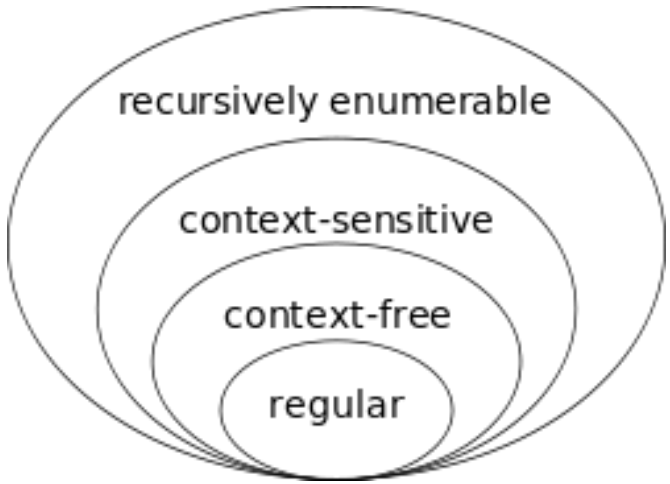




Lenguajes Regulares

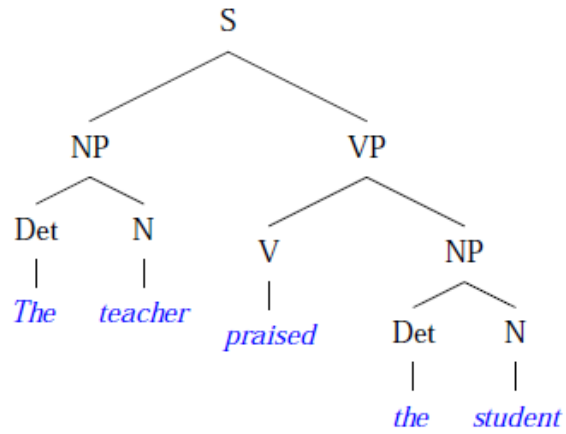
- Autómatas finitos y expresiones regulares
- Álgebra de las expresiones regulares
- Propiedades de los lenguajes regulares.

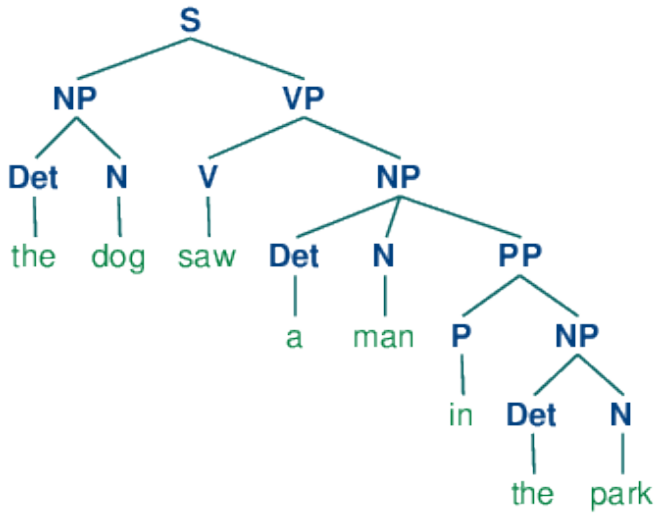




Lenguajes independientes del contexto

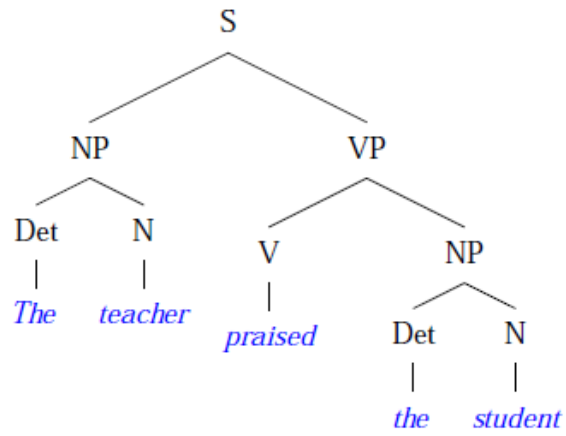
- Gramática independiente del contexto
- Árboles de derivación
- Autómatas de Pila

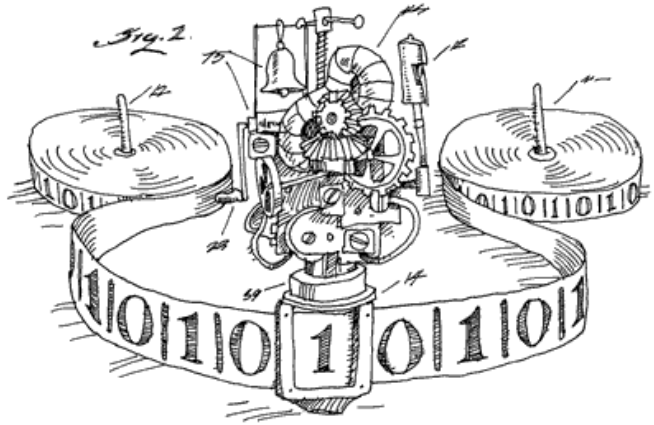




Máquinas de Turing (MT)

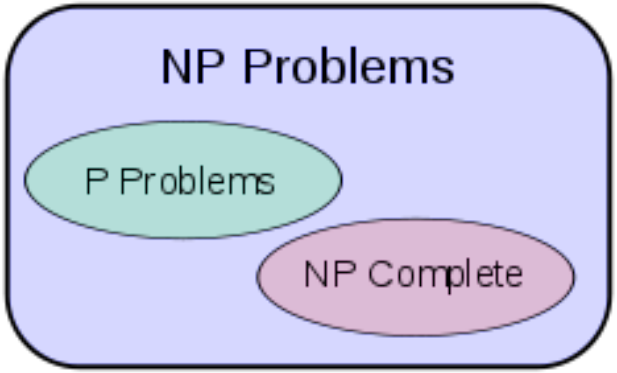
- Introducción a la MT
- Técnicas de programación para la MT
- MT restringidas
- Indecibilidad





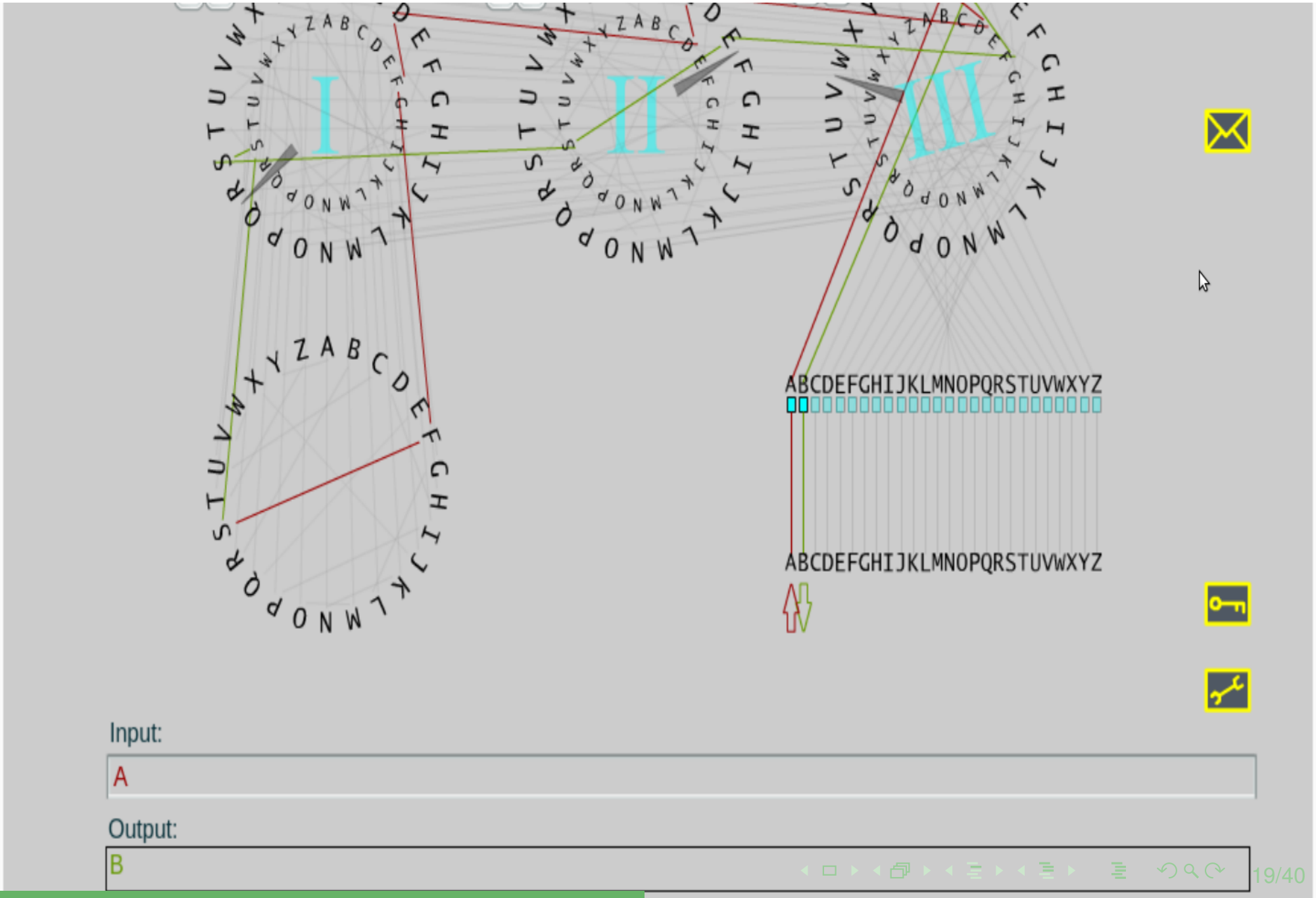
Problemas Intratables

- Las clases P y NP
- problemas NP completos
- Problemas resolubles en espacio polinómico
- Problemas PS-completos



Agenda

Proyecto Enigma

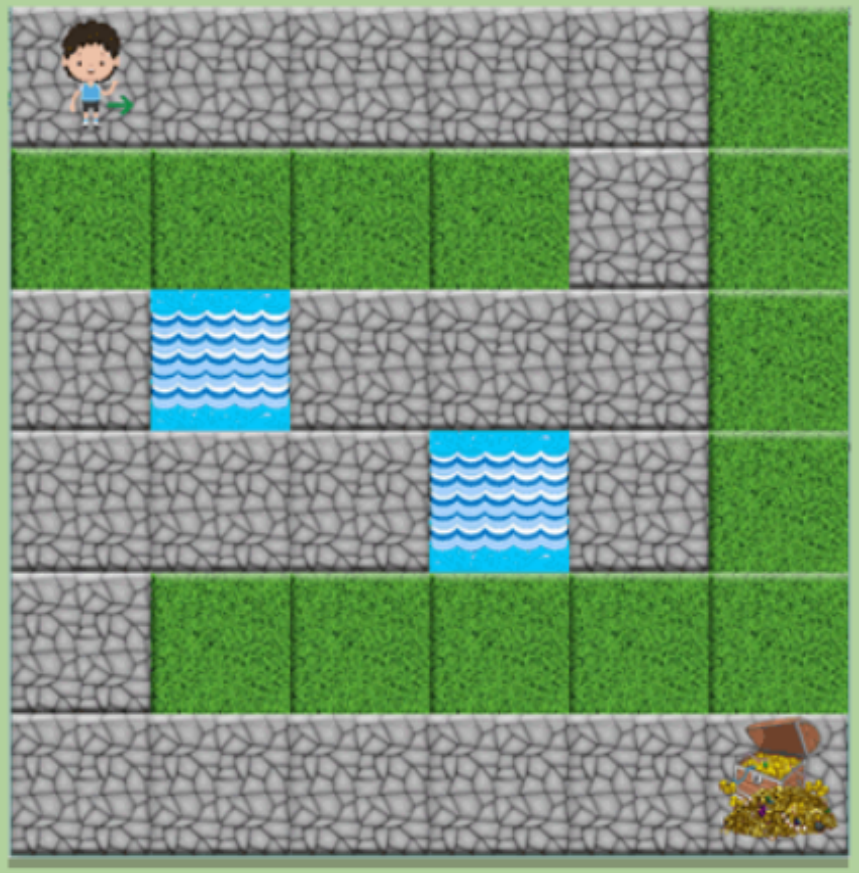


LearnToCode

PEDRO

```
codigo{
        repetir(4){
                mover;
        }
        girar(abajo);
        repetir(2){
                mover;
        }
        girar(izquierda);
        mover;
        mover;
        girar(abajo);
        mover;
        girar(izquierda);
        mover;
```

LEARN TO CODE



RUN

Autómatas y Lenguajes Formales

PROYECTO DE AUTÓMATAS JUEGO DEL AHORCADO

ALUMNOS:

- ANDRES CAMILO HERNÁNDEZ ARIAS. COD:2130284
- JULIANA ANDREA JIMENEZ BUITRAGO. COD:2140408

Profesor: Fabio Martínez

```
   +---+
   |   |
   O   |
  /|\  |
       |
       |
 =========''', '''

   +---+
   |   |
   O   |
  /|\  |
  /    |
       |
 =========''', '''

   +---+
   |   |
   O   |
  /|\  |
  / \  |
       |
 =========''']

claves = []
for sentence in (generate(grammar)):
    claves.append(sentence)

def obtenerClaveAzar (claves):
    indiceClave = ' '.join(claves[random.randint(0, len(claves) -1)])
    return indiceClave

def mostrarTablero(imagenes_juego, letrasIncorrectas, letrasCorrectas, palabraClave):
    print(imagenes_juego[len(letrasIncorrectas)])
    print ""

    print 'Letras incorrectas: '
    for letra in letrasIncorrectas:
        print letra,
    print
```

MÁQUINA DE TURING PARA DERIVAR MONÓMIOS Y AUTÓMATA FINITO DETERMINISTA ÉPSILON QUE MUESTRA EL ORDEN DE DERIVACIÓN DE UNA EXPRESIÓN

Slide Type Slide

MOTIVACIÓN

Como se aprendió en el curso de Autómatas y Lenguajes Formales, usando Máquinas de Turing y Autómatas se pueden resolver todos los problemas que un computador normal resuelve, y aunque estos problemas pueden ser más o menos complejos computacionalmente en una máquina o en un autómata, es importante clarificar el funcionamiento básico de estos temas aprendidos.

La derivación es un tema difícil para los estudiantes de grados décimo y once de los colegios de la ciudad. Entonces esta herramienta nace como una solución didáctica para estos, al poder ellos a través de esta corregir los errores que tengan a la hora de derivar monómios y dándoles instrucciones precisas a la hora de derivar expresiones más complejas.

Slide Type Sub-Slide

```
#comvierte el polinomio ingresado en unario
def comv_unario(polinomio):
    trosos_de_polinomio = polinomio.split()
    numero_1=trosos_de_polinomio[0]
    numero_2=trosos_de_polinomio[3]
    numero_1 = int(numero_1)
    numero_2 = int(numero_2)
    a = "1"
    cadena_para_tm = a*numero_1 + "0" + a*numero_2
    return cadena_para_tm
```

Slide Type Sub-Slide

```
vistaUser()
```

```
por favor ingrese el polinomio que desea derivar
"2 x ^ 3"
110111
palabra grabada en la cinta
['', '1', '1', '0', '1', '1', '1', '']
--------ejecucion de un proceso--------
----------------cinta1-----------------
['', 'x', 'x', '0', '1', '1', '', '']
--------ejecucion de un proceso--------
----------------cinta2-----------------
['', '1', '1', '1', '1', '1', '1', '0', '1', '1', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
 '', '', '', '']
proceso ejecutado con normalidad, los resultados son:
['', 'x', 'x', '0', '1', '1', '', '']
['', '1', '1', '1', '1', '1', '1', '0', '1', '1', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
 '', '', '', '']
6 x^2
```

Maquina de Turing que resuelve sistemas de ecuaciones lineales de dos variables con dos incognitas

Profesor:

Fabio Martínez Carillo

Estudiantes:

David Villabona Ardila
Juan Felipe Chacón López

Escuela de Ingeniería de Sistemas e Informática

Autómatas y Lenguajes Formales

24 de Julio del 2017

Esta es una maquina de Turing que permite encontrar el valor de la variable y para un sistema de ecuaciones lineales con variables x y y.

Para encontrar el valor de la variable y se utiliza el método matematico de reducción de incognitas.

La maquina de turing está definida en un sistema unario y los simbolos de cinta son los siguientes:

"s": Indica el signo positivo del termino

"r": Indica el signo negativo del termino

"a": Simbolo que indica donde termina una ecuacion y empieza la otra

"/": Indica división entre dos números

"i": Indica el simbolo "=" en la ecuación

"": Indica el blanco en la cinta

```
            C1[positionTape1]=apunt[1]
            C2[positionTape2]=apunt[2]
            C3[positionTape3]=apunt[3]
            C4[positionTape4]=apunt[4]
            positionTape1 = positionTape1 + MOV(apunt[5])
            positionTape2 = positionTape2 + MOV(apunt[6])
            positionTape3 = positionTape3 + MOV(apunt[7])
            positionTape4 = positionTape4 + MOV(apunt[8])
            apunt1=(apunt[0],C1[positionTape1],C2[positionTape2],C3[positionTape3],C4[positionTape4])
        print "cinta 4 "
        print C4
```

Maquina funcionando.

Llama a la función maquinaTuring e ingresa la palabra con la ecuación que se quiere resolver.

In [5]:
```
maquinaTuring("s111s1is11111as11r11is11")
```

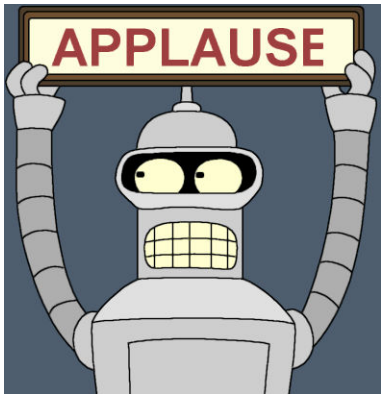
```
palabra guardada en C1
['', 's', '1', '1', '1', 's', '1', 'i', 's', '1', '1', '1', '1', '1', 'a', 's', '1', '1', 'r', '1', '1', 'i', 's',
 '1', '1', '']
cinta 4
['', 's', '1', '1', '1', '1', '1', '1', '1', '1', '/', '1', '1', '1', '1', '', '', '', '', '', '', '', '', '', '', '',
```

Agenda

Informática: información automática

Teoría de la computabilidad

- Origen: toda clase de problemas pudiese ser resuelto desde un esquema general: **máquina automática**
  - ¿Qué pueden hacer los computadores?
  - ¿Cuáles son los límites inherentes de las máquinas?



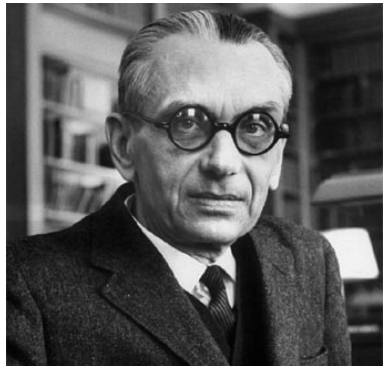
D. Hilbert: Entscheidungsproblem (1928)

Problema de decisión

- ¿Las matemáticas son completas?. En el sentido que pueda probarse cada aseveración matemática
- ¿Las matemáticas son consistentes? . Puede probarse cada aseveración y su negación
- ¿Las matemáticas son decidibles? Existe un método que pueda aplicarse y que determine la aseveración

Objetivo: crear un sistema matemático completo y consistente.

K Godel: Teorema de la incompletitud (1931)



“ Todo sistema de primer orden que contenga teoremas de la aritmética y cuyo conjunto de axiomas sea recursivo ***no es completo****”*

- Si los axiomas de dicha teoría no se contradicen entre sí, entonces existen enunciados que no pueden probarse ni refutarse a partir de ellos.
- La conclusión del teorema se aplica siempre que la teoría aritmética en cuestión sea recursiva

Church λ-definible (1936)

λ-definible

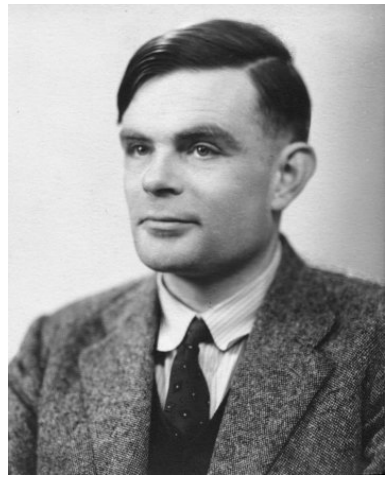
- Propone una función calculable
- La demostración de teoremas se convierte en una transformación de una cadena de simbolos según un conjunto de reglas
- El sistema fue inconsistente

Kleene

Clausura de Kleene

- Una operación unaria que se aplica sobre un conjunto de cadenas de caracteres o un conjunto de símbolos
- Demuestra la equivalencia entre las funciones λ y las funciones de Godel

A. Turing (1936)



Máquinas abstractas

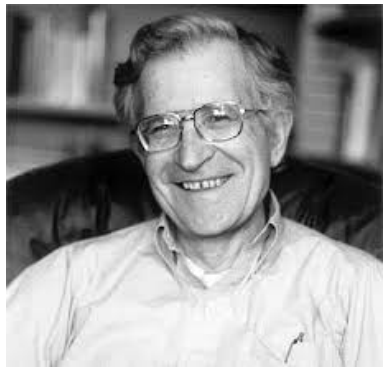
- Tesis de Turing: funciones calculables mediante un algoritmo
- Utilizó el concepto de *máquina* para demostrar que hay funciones no calculables
- La tesis de Turing es equivalente a la de Church

En los 60

- Se separan los conceptos de las implementaciones
- Aparecen los lenguajes de alto nivel como Fortran
- Se estudian algoritmos independientes del ordenador
- Aparece la necesidad de traducir lenguajes de alto nivel a lenguaje de Máquina

Desde los años 40 se ha intentado imitar funciones del cerebro biológico. Se han desarrollado máquinas capaces de aprender y reproducir funciones o comportamientos.
Estas máquinas se llaman Redes Neuronales.

Chomsky



Propone tres modelos para la descripción de lenguajes que ayudaron al desarrollo de la programación

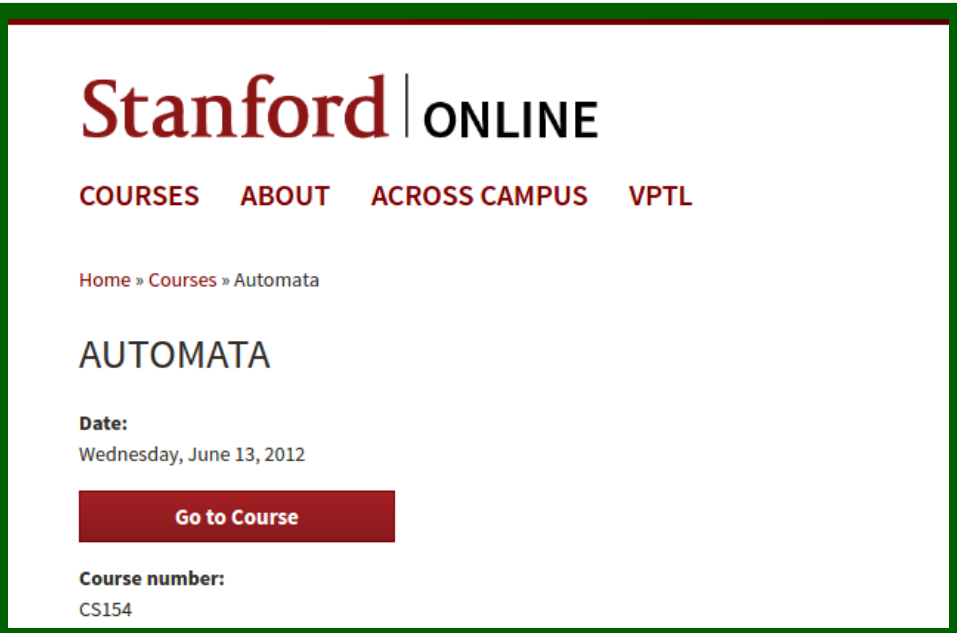
- Define la diferencia entre autómatas y gramáticas

Agenda

Libros

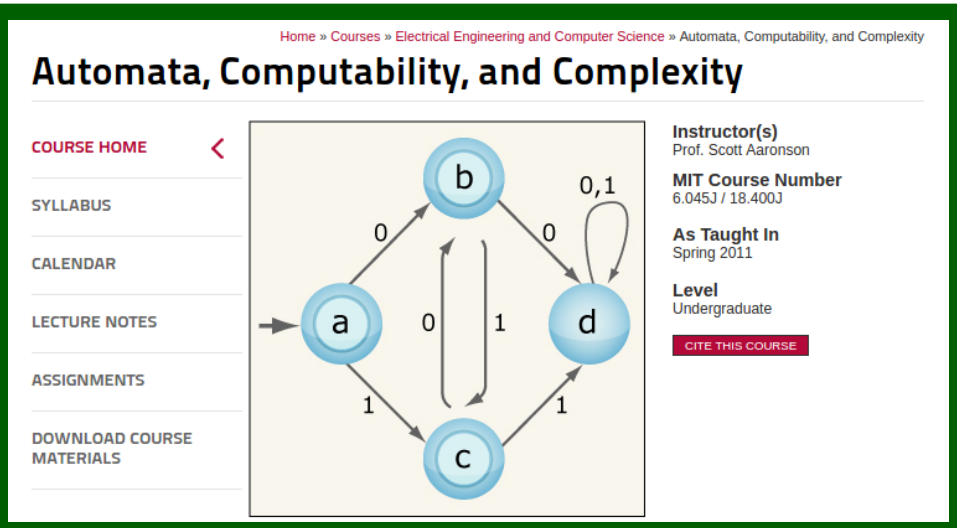
- KELLEY, Dean. Teoría de autómatas y lenguajes formales, Prentice Hall.
- HOPCROFT, J.E., and J. D. ULLMAN. Teoría de autómatas, lenguajes y computación. Pearson 2007
- SUDKAMP, T. Languages and Machines, Addison-Wesley Publishing Company, Inc, Reading, Mass, 1988.
- BRENA RAMON. Autómatas y lenguajes. 2003

Cursos e información en línea



http://online.stanford.edu/course/automata

Cursos e información en línea



https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-045j-automata-computability-and-complexity-spring-2011/

Muchas gracias por su atención