Autómatas y Gramáticas
(Ex Teoría de Compiladores)

Docentes:

Dr. Ing. Alfredo Iglesias

Mg. Ing. Nora Costa

Prerrequisitos

- Para aprovechar este curso, los estudiantes deberían conocer previamente temas como matemática discreta, temas como grafos, árboles y lógica.
- Programación en Phyton y estar familiarizados con las estructuras de datos más comunes, pilas, colas. También conceptos de recusión.

Bibliografía

- John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, **"TEORÍA DE AUTÓMATAS, LENGUAJES Y COMPUTACIÓN".** *Pearson Addison Wesley. (Tercera Edición).*

- A. Aho, R.Sethi, J. Ullmam **"COMPILADORES PRINCIPIOS, TECNICAS Y HERRAMIENTAS".** *Pearson Addison Wesley. (Primera y Segunda Edición).*

- Dean Kelley "**TEORÍA DE AUTÓMATAS Y LENGUAJES FORMALES"** *Prentice Hall (Segunda Edición).*

Cronograma 2020

- 10/08 Introducción a la materia. Introducción a Autómatas Finitos.
- 17/08 Feriado: Paso a la Inmortalidad del General José de San Martín.
- 24/08 Autómatas Finitos Deterministas. Autómatas Finitos No Deterministas. Aplicaciones.
- 31/08 Lenguajes y expresiones regulares. Operaciones con expresiones regulares Autómatas Finitos y Expresiones Regulares. Aplicaciones

- 07/09 Trabajo Práctico Nº 1: Implementación con Python para validación de datos mediante Expresiones Regulares.
- 14/09 Mesas Especiales.
- 21/09 Feriado: Día del Estudiante.
- 28/09 Lenguajes y Gramáticas independientes del contexto. Derivaciones. Árboles Sintácticos. Gramáticas Ambiguas.
- 05/10 Analizadores Sintácticos. Autómatas con pila.
- 12/10 Feriado: Día del Respeto a la Diversidad Cultural.

- 19/10 Trabajo Práctico Nº 2: Programación de analizadores sintácticos en Phyton.

- 26/10 Máquinas de Turing. Definición formal. Diagrama de estados. Lenguajes asociados. Restricciones. Máquinas de Turing no deterministas. Implementación de una máquina de Turing a partir de un autómata finito determinista.

- 02/11 Trabajo Práctico Nº 3: Autómatas de estados finitos y Máquinas de Turing.

- 09/11 Trabajo Práctico Nº 4: Análisis de datos con Phyton sobre una base de datos de tráfico de wifi.

Requisitos de Regularidad

Para obtener la "regularidad" se requiere la presentación de los trabajos prácticos 1, 2 y 3 dentro de las dos semanas de haber sido propuestos. La regularidad en la materia tiene una duración de dos años.

Fechas límites para la presentación de Prácticos:

Trabajo Práctico Nº 1: antes del 21/09

Trabajo Práctico Nº 2: antes del 02/11

Trabajo Práctico Nº 3: antes del 23/11

Aprobación de la Asignatura

Obtenida la regularidad, deberá inscribirse en uno de los turnos de exámenes que fije la Facultad.

El examen final consistirá en la presentación del Trabajo Práctico Nº 4 en forma sincrónica (google meet) y eventualmente preguntas sobre como fue desarrollado.

Realización de Trabajos Prácticos

Los Trabajos Prácticos pueden realizarse individualmente, o en grupos de no más de tres estudiantes por grupo.

La conformación de estos grupos puede cambiar de un trabajo práctico a otro.

Tengan en cuenta que la condición de regularidad es individual, es decir solo la alcanzan aquellos estudiantes que presentaron y aprobaron los prácticos 1,2 y 3, independientemente del grupo con el que los realizaron.

Introducción

La teoría de autómatas es el estudio de dispositivos de cálculo abstractos, es decir, de las “máquinas”.

Antes de que existieran las computadoras, en la década de 1930, A. Turing estudió una máquina abstracta que tenía todas las capacidades de las computadoras de hoy día.

(ver película “The Imitation Game” 2014)

https://www.youtube.com/watch?v=0DFqSmdbsUM

En las décadas de los años 1940 y 1950, una serie de investigadores estudiaron las máquinas más simples, las cuales todavía hoy denominamos “autómatas finitos”. Originalmente, estos autómatas se propusieron para modelar el funcionamiento del cerebro y, posteriormente, resultaron extremadamente útiles para otros propósitos.

- En 1969, S. Cook amplió el estudio realizado por Turing sobre lo que se podía y no se podía calcular.
- Hay problemas que, en principio, pueden resolverse, pero que en la práctica consumen tanto tiempo que las computadoras resultan inútiles para todo excepto para casos muy simples del problema. Este último tipo de problemas se denominan "**insolubles**" o "**NP-difíciles** ".

- Los autómatas finitos son un modelo útil para muchos tipos de hardware y software, los tipos más importantes son:

- Software para diseñar y probar el comportamiento de circuitos digitales.

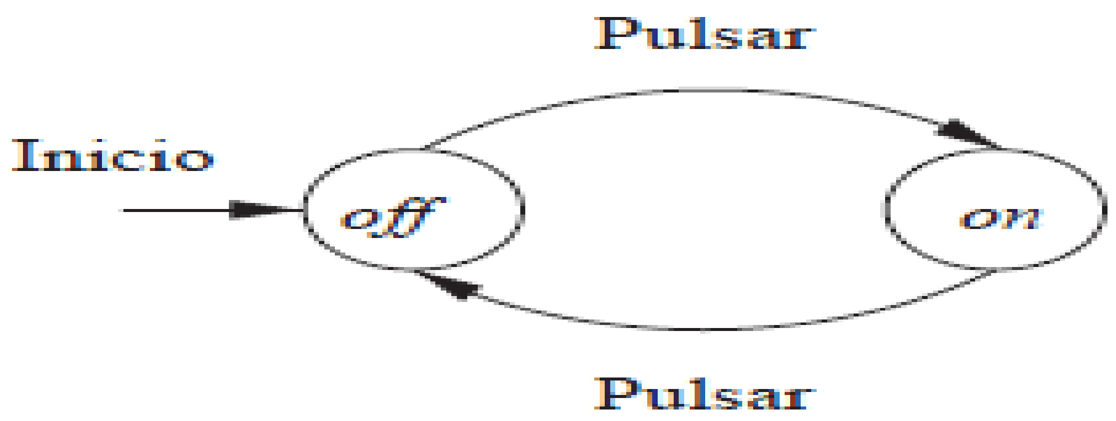
- El “analizador léxico” de un compilador típico, es decir, el componente del compilador que separa el texto de entrada en unidades lógicas (tokens), tal como identificadores, palabras clave y signos de puntuación.

- Software para explorar cuerpos de texto largos, como colecciones de páginas web, o para determinar el número de apariciones de palabras, frases u otros patrones.

- Software para verificar sistemas que tengan un número finito de estados diferentes, tales como protocolos de comunicaciones o protocolos para el intercambio seguro de información.

Ejemplo

- El autómata finito más simple puede ser un interruptor de apagado/encendido (posiciones on/off). El dispositivo **recuerda** si está en el estado encendido ("on") o en el estado apagado ("off"),



Conceptos fundamentales

Alfabetos

Un alfabeto es un conjunto de símbolos **finito y no vacío**. Convencionalmente, utilizamos el símbolo ∑ para designar un alfabeto. Entre los alfabetos más comunes se incluyen los siguientes:

- 1. ∑ = {0,1}, el alfabeto binario.
- 2. ∑ = {a,b, . . . , z}, el conjunto de todas las letras minúsculas.
- 3. ∑ = {0,1, . . . ,9}, el conjunto de los dígitos del 0 al 9.
- 4. El conjunto de todos los caracteres ASCII o el conjunto de todos los caracteres ASCII imprimibles.

  ASCII (*American Standard Code for Information Interchange*)

Cadenas de caracteres

Una cadena de caracteres (que también se denomina en ocasiones palabra) es una secuencia finita de símbolos seleccionados de algún alfabeto. Por ejemplo, 01101 es una cadena del alfabeto binario $\sum = \{0,1\}$. La cadena 111 es otra cadena de dicho alfabeto.

La cadena vacía

La cadena vacía es aquella cadena que presenta cero apariciones de símbolos. Esta cadena, designada por **ε**, es una cadena que puede construirse en cualquier alfabeto.

Longitud de una cadena

Suele ser útil clasificar las cadenas por su longitud, es decir, el número de posiciones ocupadas por símbolos dentro de la cadena. Por ejemplo, 01101 tiene una longitud de 5.

La notación estándar para indicar la longitud de una cadena w es $|w|$. Por ejemplo, $|011| = 3$

Potencias de un alfabeto

Si $\sum$ es un alfabeto, podemos expresar el conjunto de todas las cadenas de una determinada longitud de dicho alfabeto utilizando una notación exponencial.

Definimos para que sea el conjunto de las cadenas de longitud k, tales que cada uno de los símbolos de las mismas pertenece a $\sum$

Observe que $\Sigma^0 = \varepsilon$ independientemente de cuál sea el alfabeto Σ es decir, ε es la única cadena cuya longitud es 0.

Si $\Sigma = \{0,1\}$, entonces:

$$\Sigma^0 = \varepsilon$$
$$\Sigma^1 = \{0,1\}$$
$$\Sigma^2 = \{00,01,10,11\}$$
$$\Sigma^3 = \{000,001,010,011,100,101,110,111\}$$

El conjunto de todas las cadenas de un alfabeto Σ se designa mediante Σ^* . Por ejemplo:

$$\Sigma^* = \{ \varepsilon ,0,1,00,01,10,11,000, . . .\}$$

Expresado de otra forma:

$$\Sigma^* = \Sigma 0 \cup \Sigma 1 \cup \Sigma 2 \cup \cdots$$

En ocasiones, desearemos excluir la cadena vacía del conjunto de cadenas. El conjunto de cadenas no vacías del alfabeto $\sum$ se designa como $\sum +$. Por tanto, dos equivalencias apropiadas son:

- $\sum += \sum 1 \cup \sum 2 \cup \cdots$
- $\sum * = \sum + \cup \{ \boldsymbol{\varepsilon} \}$.

Concatenación de cadenas

Sean x e y dos cadenas. Entonces, xy denota la concatenación de x e y, es decir, la cadena formada por una copia de x seguida de una copia de y. Dicho de manera más precisa, si x es la cadena compuesta por i símbolos

- $x = a_1a_2 \cdots a_i$ e y es la cadena compuesta por j símbolos $y = b_1b_2 \cdots b_j$, entonces xy es la cadena de longitud
- $i + j$: $xy = a_1a_2 \cdots a_ib_1b_2 \cdots b_j$.

Potencia:

$$w^n = \begin{cases} \varepsilon & n = 0 \\ ww^{n-1} & n > 0 \end{cases}$$

Si w = 122 sobre el alfabeto Σ = { 1, 2 }

$w^0 = \varepsilon$

w^1=122

w^2=122122

w^3=122122122