DESARROLLO DE UN ENTORNO DE SIMULACIÓN PARA AUTÓMATAS DETERMINISTAS

Development of a simulation environment for deterministic automatas

ÁLVARO ANGEL OROZCO GUTIÉRREZ
Doctor en Bioingeniería
Director Grupo de Investigación en Control e Instrumentación
Profesor Titular
Universidad Tecnológica de Pereira
aaog@utp.edu.co

MAURICIO HOLGUÍN LONDOÑO
Ingeniero Electricista, M.Sc (C)
Profesor Transitorio
Estudiante Maestría en Ingeniería Eléctrica.
Universidad Tecnológica de Pereira
ma_hol@ohm.utp.edu.co

RESUMEN
Se muestra el poderío matemático y generalidad de la Máquina de Turing entre las máquinas abstractas equivalentes a la jerarquía de lenguajes formales que desarrolló Noam Chomsky en su obra Teoría de las Gramáticas Transformacionales, por medio del desarrollo de un simulador de autómatas; que permite representar el funcionamiento de un reconocedor de lenguajes que determina si una palabra, cadena finita de símbolos de un alfabeto, pertenece o no a un lenguaje dado. Se enmarca como herramienta pedagógica que permite mostrar la generalidad de la máquina de Turing al abarcar el conjunto de los autómatas finitos y de pila.

PALABRAS CLAVES: Autómatas, Máquina de Estados Finitos, Máquina de Turing.

ABSTRACT
It shows the mathematics power and generality of Turing Machine respect to abstract machines equivalents to the hierarchy of formal languages developed by Noam Chomsky in his book Theory of Transformational Grammars, by developing a automata simulator, which allows represent the behavior of a language recognizer that determines whether a word, string of symbols from a finite alphabet, belongs to a given language. It is framed as a teaching tool that shows the generality of the Turing Machine to cover all of the finite automatas and push down machines.

KEYWORDS: *Automatas, Finite State Machine, Turing Machine.*

1. INTRODUCCIÓN

Turing propuso en los años treinta un modelo de máquina abstracta [1], como una extensión de los autómatas finitos, que resultó ser de una gran simplicidad y poderío a la vez. La máquina de Turing es particularmente importante porque es la más poderosa de todas las máquinas abstractas conocidas.

Se ha tratado de encontrar otros modelos de máquinas o formalismos que posean una mayor generalización que las Maquinas de Turing, en el mismo sentido que las Máquinas de Turing son más generales que los Autómatas Finitos y los Autómatas de Pila [2]; (se dice que un tipo de máquina MA tiene una mayor generalidad que un tipo MB cuando el conjunto de lenguajes aceptados por alguna máquina en MB es un subconjunto propio de los aceptados por MA), pero todos los intentos han sido infructuosos al encontrar que dichas extensiones son equivalentes en poder de cálculo a la Máquina de Turing original [3]; muchas de estas extensiones dotan de mayor flexibilidad al diseño de una Máquina de Turing que resuelve un problema en particular.

A. Turing propuso, en su llamada Tesis de Turing, que la noción de una máquina de Turing es una idealización matemática útil para probar que ciertas tareas no son automatizables o que ciertas funciones no son computables, una función es computable si y sólo si hay una máquina de Turing que la implementa, y que no podrá haber una máquina abstracta que calcule algo que la Máquina de Turing no pueda calcular [4].

Todos los sistemas modernos de cómputo, pueden ser modelados utilizando la abstracción matemática de la Máquina de Turing, pues permite describir mediante un conjunto simple de operaciones todas las funciones computables [5, 6].

Un reto grande consiste en mostrar pedagógicamente el sentido por medio del cual una máquina dada posee, o no, mayor poderío. Aunque existe amplia documentación relacionada con la teoría de los autómatas, no se cuenta con una herramienta simple, y a la vez clara y directa, que permite enseñar a los alumnos la jerarquía en los lenguajes aceptados por un autómata y por ende la jerarquía de estos últimos. El objetivo primordial es desarrollo de un simulador de autómatas que permita

Fecha de Recepción: 8 de julio de 2009
Fecha de Aceptación: 13 de Agosto de 2009

afrontar de una forma pedagógica todos los retos planteados previamente.

2. CONTENIDO

2.1. LENGUAJES

Un autómata es un dispositivo que manipula cadenas de símbolos que se le presentan a su entrada, produciendo otras cadenas de símbolos (palabras) a su salida. Las clases de lenguajes son conjuntos de lenguajes que comparten una cierta propiedad dada; Chomsky propuso una jerarquía de lenguajes [7], donde las clases más complejas incluyen a las más simples.

- Lenguajes Regulares: la clase más pequeña, e incluye a los lenguajes más simples, se llaman así porque sus palabras contienen regularidades o repeticiones de los mismos componentes, estos lenguajes son finitos y pueden ser resultado de la unión o concatenación de otros lenguajes regulares.

- Lenguajes Libres de Contexto: incluyen a los Lenguajes Regulares. Estos lenguajes son generados por las gramáticas Libres de Contexto donde un símbolo no-terminal puede ser sustituido por una cadena de símbolos terminales y/o no-terminales, sin tener en cuenta el contexto en el que ocurra.

- Lenguajes Recursivamente Enumerables: incluyen a los Libres de Contexto y por lo tanto a los Lenguajes Regulares. Es un lenguaje formal para el cual existe una máquina de Turing que acepta y se detiene con cualquier cadena del lenguaje; son llamados parcialmente decidibles o Turing Computables.

2.2. MÁQUINA DE TURING

La máquina de Turing es una abstracción matemática y se representa como se muestra en la figura 1. El hecho que se le denomine "máquina" se debe a que su funcionamiento puede ser descrito en términos de operaciones menores muy sencillas que sugieren una implementación real simple, lo que ha motivado que existan muchas versiones prácticas de la misma.

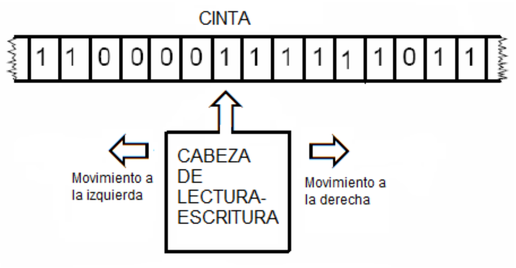

Figura 1. Máquina de Turing

Existen diversas "variedades" de una máquina de Turing, pero la más simple puede ser descrita diciendo que es cualquier dispositivo que cumple las siguientes condiciones:

- Tiene una cinta sobre la que un cabezal de lectura/escritura puede hacer desplazamiento a izquierda y derecha. La cinta contiene una serie de celdas y en cada una de ellas se puede escribir un símbolo de un conjunto finito; este conjunto de símbolos se denomina el alfabeto de la máquina. En principio todas las celdas que no se hayan escrito antes contienen un carácter especial nulo o vacío. La cinta puede contener tantas celdas a derecha e izquierda del cabezal como sean necesarias para el funcionamiento de la máquina.

- El cabezal puede moverse a derecha (R) o izquierda (L) de su posición actual, así como leer el contenido de una celda o escribir en ella cualquier carácter de su alfabeto.

- Existe un registro que almacena el estado de la máquina. El número de estados posibles es finito y no se exige ningún estado especial con el que sea iniciada la máquina.

- Existe una tabla de acción, que contiene las instrucciones de lo que hará el autómata. Estas instrucciones representan en cierta forma el "programa" de la máquina. La ejecución de cada instrucción de la tabla de acción incluye cuatro pasos:

  - Leer un carácter en la posición actual.
  - Escribir un nuevo símbolo en esta posición, puede ser el mismo que había. El símbolo a escribir es alguno del alfabeto de la máquina y depende del carácter leído y del estado actual.
  - Desplazar el cabezal una celda a derecha o izquierda (R/L); en algunos modelos el desplazamiento puede ser nulo.
  - Decidir cuál será el nuevo estado en función del carácter que se acaba de leer y del estado actual. Si la tabla de acción no contiene ninguna correspondencia con el estado actual y el símbolo leído, entonces la máquina detiene su funcionamiento: *halt*.

Formalmente, una Máquina de Turing es un quíntuplo $(K, \Sigma, \Gamma, \delta, s)$, donde:

- K : es un conjunto de estados
- Σ : es el alfabeto de entrada
- Γ : es el alfabeto de la cinta
- s : es el estado inicial $s \in K$
- δ : función de transición de estados

2.3. AUTÓMATA DE PILA

Un autómata de pila, AP, reconoce los lenguajes libres de contexto [8]. Es una máquina abstracta que se puede representar como una máquina de Turing que sólo puede leer desde una cinta y que puede guardar en una pila. Su capacidad de procesamiento es limitada debido a las siguientes restricciones sobre las posibles operaciones con la cinta y la pila:

- La cinta se desplaza en un sólo sentido y su cabeza sólo puede leer.
- La pila está limitada en un extremo por definición, cuando se lee un elemento de la pila, éste desaparece o se saca, y cuando se escribe en la pila se introduce un elemento.

En cualquier caso, si se vacía la cinta el autómata se detiene. Un autómata de pila se puede representar intuitivamente según la figura 2.

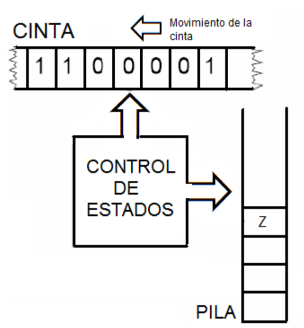

Figura 2. Autómata de Pila

Un autómata de pila es un séxtuplo $(K, \Sigma, \Gamma, \Delta, s, F)$, donde:

- K : conjunto de estados.
- Σ : alfabeto de entrada
- Γ : alfabeto de la pila
- s : es el estado inicial $s \in K$
- F : es un conjunto de estados finales, $F \subseteq K$
- Δ : es la relación de transición

2.4. AUTÓMATA FINITO

Un autómata finito es una estructura matemática que representa un sistema o máquina abstracta, reconoce los lenguajes regulares [9], puede representarse intuitivamente por una cinta y una cabeza de lectura, como se muestra en la figura 3.

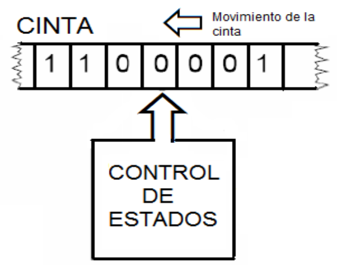

Figura 3. Autómata Finito.

La cinta, que se extiende infinitamente hacia la derecha, está dividida en celdas, cada una de las cuales es capaz de almacenar un sólo símbolo de un cierto alfabeto. La máquina es capaz de leer los símbolos de esta cinta de izquierda a derecha por medio de un cabezal de lectura.

Cada vez que se lee un símbolo, el cabezal de lectura se mueve a la siguiente celda a la derecha y la máquina efectúa un cambio de estado o transición. Esta transición está determinada por el mecanismo de control (que contiene un número finito de estados), programado para conocer cuál debe ser el nuevo estado, que dependerá de la combinación del estado actual y el símbolo de entrada leído.

Los autómatas finitos pueden considerarse como mecanismos aceptadores o reconocedores de palabras. De manera informal se dice que un autómata finito aceptará una palabra de entrada si comenzando por el estado inicial y estando la cabeza de lectura apuntando al primer símbolo de la cadena, la máquina alcanza un estado final o de aceptación después de leer el último símbolo de la cadena.

Formalmente un autómata finito se define como un quíntuplo $(K, \Sigma, \delta, s, F)$ donde:

- K : conjunto finito de estados
- Σ : conjunto finito de símbolos de entrada
- δ: función de transición de estados, que se define como δ: $K \times \Sigma \rightarrow K$ si el autómata es determinístico
- s : estado inicial; $s \in K$
- F : conjunto de estados finales o estados de aceptación; $F \in K$

3. ENTORNO DE SIMULACIÓN PARA AUTÓMATAS DETERMINISTAS

Para el presente artículo se desarrolló un simulador de autómatas deterministas; como una herramienta pedagógica que permite mostrar como la máquina de Turing abarca el conjunto de todas las máquinas abstractas.

A continuación se muestran las capacidades y limitaciones de cada máquina por medio de un ejemplo que confronta la teoría en cuestión y permite resaltar la generalidad y poderío de la Máquina de Turing, mostrando el funcionamiento, validez y confiabilidad del entorno de simulación desarrollado.

3.1. Máquina de estados finitos

$\{a^2b^2\}$ es un tipo de lenguaje formal, pues sus palabras contienen regularidades de los mismos componentes.

El funcionamiento de los autómatas finitos consiste en ir pasando de un estado a otro, a medida que van recibiendo caracteres de la palabra de entrada, como se muestra en las figuras 4 y 5, donde al leer el primer caracter toma como estado actual el estado 1 y al leer el segundo caracter pasa al estado 3.

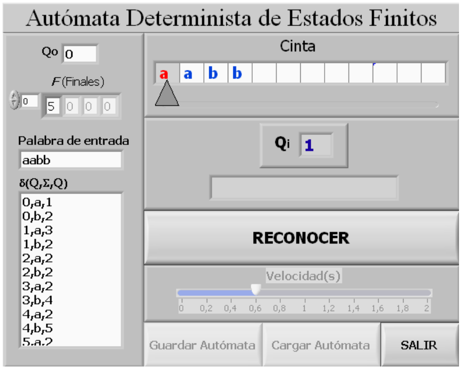

Figura 4. Autómata reconocedor de la palabra a^2b^2 en el estado 1.

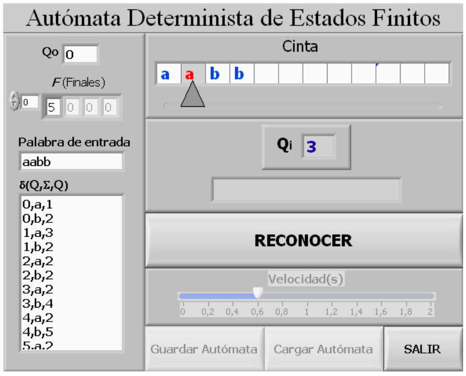

Figura 5. Autómata reconocedor de la palabra a^2b^2 en el estado 3.

El autómata finito determinista acepta la cadena $aabb$, figura 6, porque al leer todos los caracteres de dicha palabra de entrada, y por la secuencia de transiciones, se conduce desde el estado inicial al estado final.

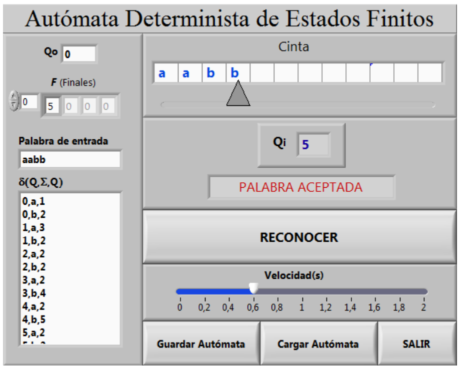

Figura 6. Aceptación del la palabra $aabb$.

La trayectoria seguida para la palabra $aabb$ consiste en la secuencia de estados: q_1, q_3, q_4 y q_5. Los estados son el único medio del que disponen los autómatas finitos para recordar qué caracteres se han leído hasta el momento, son máquinas de memoria limitada. Para el lenguaje $\{a^nb^n\}$ no es posible construir un autómata finito que lo acepte, ni representarlo por una expresión regular, pues al terminar de leer el grupo de $a's$ el autómata debe recordar cuántas encontró, para poder comparar con el número de $b's$. Como la cantidad de $a's$ que puede haber en la primera mitad de la palabra no es constante, dicha cantidad no puede recordarse con una memoria fija, como es la de los autómatas finitos.

3.2. Máquina de pila

$\{a^nb^n\}\cup\{b^na^n\}$ es un lenguaje libre de contexto, el cual forma una clase de lenguaje más amplia que los lenguajes regulares, de acuerdo con la Jerarquía de Chomsky [7]. Como los autómatas finitos no son suficientemente poderosos para aceptar los lenguajes libres de contexto se les agregó una memoria en forma de pila que incrementa su poder de cálculo. La pila tiene un alfabeto propio, que puede o no coincidir con el alfabeto de la palabra de entrada.

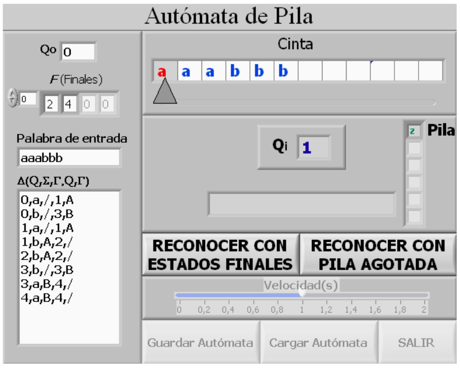

Figura 7. Pila vacía.

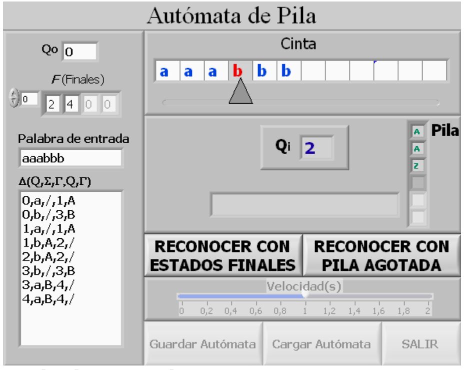

Figura 8. Pila almacenando caracteres.

Como se muestra en las figuras 7 y 8 al iniciar la operación la pila se encuentra vacía (Z=carácter de pila vacía), durante la operación la pila almacena o borra caracteres, según lo indiquen las transiciones ejecutadas.

Para diseñar el autómata de pila que acepta exactamente el lenguaje con palabras de la forma $\{a^n b^n\} \cup \{b^n a^n\}$, se utilizó la pila como contador para recordar la cantidad de $a\acute{}s$ que se consumen, y luego confrontar con la cantidad de $b\acute{}s$, y los estados para memorizar las situaciones de estar consumiendo a o b.

Al final de la operación de aceptación de la palabra, la pila está vacía, se han leído todos los caracteres de la palabra de entrada y la secuencia de transiciones conduce desde el estado inicial al estado final, como se muestra para la palabra $aaabbb$ en la figura 9.

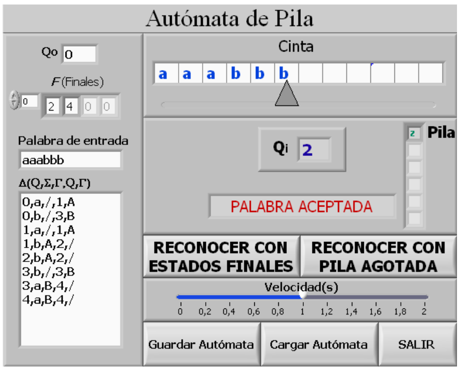

Figura 9. Aceptación de la palabra $aaabbb$.

Las gramáticas libres de contexto (GLC) deben su nombre a una comparación con las gramáticas sensitivas al contexto, donde una regla genera un símbolo cuando se encuentra rodeado por un contexto. Las gramáticas sensitivas al contexto son estrictamente más poderosas que las gramáticas libres de contexto; un ejemplo es el lenguaje de las cadenas de la forma $a^n b^n c^n$, para el que no existe una gramática libre de contexto por lo cual no es posible implementarlo en un autómata de pila.

3.3. Máquina de Turing

Para probar que hay lenguajes que no son libres de contexto, pero que pueden ser aceptados por una máquina de Turing (MT), se trabaja con el lenguaje $a^n b^n c^n$, que no es un lenguaje libre de contexto [8].

La palabra de entrada, $aabbcc$, está escrita inicialmente en la cinta. Como la cinta es infinita, inicialmente toda la parte de la cinta a la derecha de la palabra de entrada está llena del caracter blanco, figura 10.

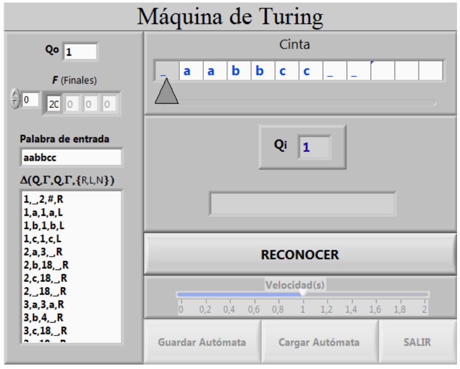

Figura 10. Ubicación de la palabra de entrada $aabbcc$ en la cinta.

Por definición, al iniciar la operación de la MT, la cabeza lectora está posicionada en el carácter a la izquierda de la palabra de entrada, $(\_)$, el cual es el cuadro más a la izquierda de la cinta figura 10.

La estrategia para el funcionamiento de esta MT consiste en recorrer la palabra, descontando en cada una de ellas una a, una b y una c y se reemplazan por un caracter "_", como se muestra en la figura 11. En la MT la cabeza lectora es de lectura y escritura, por lo que la cinta es modificada en curso de ejecución. Además, en la MT la cabeza se mueve bidireccionalmente, izquierda y derecha, por lo que pasa repetidas veces sobre un mismo segmento de la cinta.

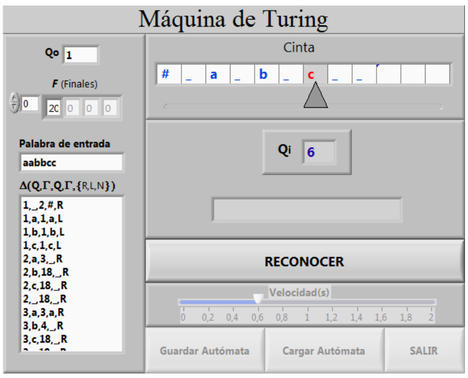

Figura 11. Funcionamiento de la MT como reconocedora del lenguaje $a^n b^n c^n$

Cuando ya no encuentra ninguna a, b o c en algún recorrido, si queda alguna de las otras dos letras la palabra no es aceptada; en caso contrario se llega a *halt*, detiene la operación de la MT, y se acepta la cadena de entrada como se muestra en la figura 12 para la palabra $aabbcc$, donde se observa que el único criterio para que la palabra de entrada sea aceptada es que se llegue a *halt* en algún momento, independientemente del contenido final de la cinta, el cual es visto como basura.

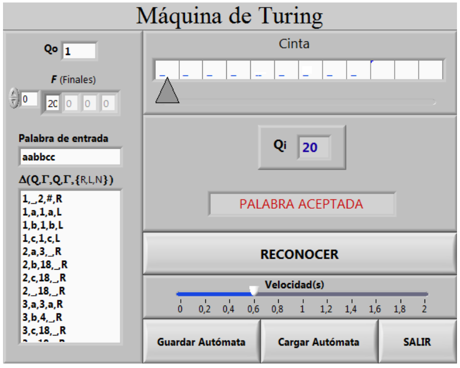

Figura 12. Aceptación de la cadena de entrada $aabbcc$.

4. CONCLUSIONES

Los autómatas han ido materializándose paulatinamente hasta convertirse en herramientas cada vez más participativas; aportando, con validez, en la realización de cálculos. En esta evolución de la materialización de los procedimientos se ha llegado a la construcción de máquinas con capacidad para recibir información y ajustar su comportamiento de acuerdo con la información recibida, máquinas en las que cada vez intervienen menos, de forma directa, las personas; siendo de gran importancia dar a conocer de forma pedagógica dichas maquinas abstractas: Autómatas Finitos, Máquina de Pila y Máquina de Turing; con sus respectivos principios, utilidades, limitaciones y jerarquías.

El entorno de simulación de autómatas deterministas desarrollado permite representar el funcionamiento de un reconocedor de lenguajes, determinando si una palabra, pertenece o no a un leguaje dado, a partir de la definición de alfabeto, estados y función de transición; como una herramienta pedagógica que enmarca el poderío matemático y generalidad de la Máquina de Turing entre las máquinas abstractas equivalentes a la jerarquía de lenguajes formales que desarrolló Noam Chomsky.

Después del trabajo realizado sobre las Máquinas de Turing se puede concluir que el tema en cuestión es bastante amplio por lo cual queda por analizar y estudiar más a fondo temas como las extensiones a la Máquina de Turing y el hecho de que puede probarse matemáticamente que para cualquier programa de computador es posible crear una máquina de Turing equivalente, lo cual es el fundamento de teorías como la computabilidad y la complejidad.

5. AGRADECIMIENTOS

Los autores agradecen a los estudiantes Carolina González Naranjo y Cesar Augusto Montoya Román del programa de Ingeniería Eléctrica de la Universidad Tecnológica de Pereira por toda su colaboración en la elaboración de algoritmos y durante el proceso de validación de la herramienta de simulación de autómatas deterministas.

6. BIBLIOGRAFÍA

[1] Turing, A. *On computable numbers with an application to the Entscheidungs-problem*, Proc. London Math.

[2] Manna, Z. *Mathematical Theory of Computation*, McGraw Hill, 1974.

[3] Lewis, Ch; Papadimitriou, H. *Elements of the Theory of Computation*, Prentice Hall, 1981.

[4] Linz, P. *An Introduction to Formal Languages and Automata*, D. C. Heath and Company, 1990.

[5] Anderson, James A. *Automata Theory with Modern Applications*, Cambridge University Press, United States, 2006.

[6] Avendaño, Luis Enrique. Reconocimiento de Patrones. Universidad Tecnológica de Pereira, 1990.

[7] Hhomsky, N. *Aspects of The Theory Of Syntax*, Cambridge, Mit Press, 1965.

[8] Sipser, M. *Introduction to the Theory of Computation*, PWS Pub. Co, 1997.

[9] Hopcroft, John E. *Introduction to Automata Theory, Languages, and Computation*. Pearson Education S.A., New York, 2000.