

NOMBRE: Nestor Adolfo Guillen Ramírez

GRADO: 5to computación

MATERIA: Producción

NUM de páginas: 8 Páginas

Temas de Investigación

**1. Teoría de autómatas**

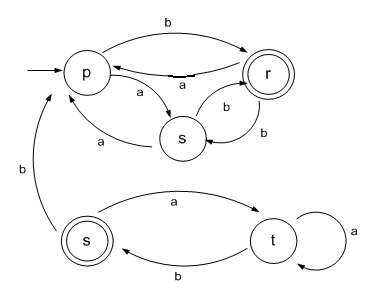
Un autómata finito determinista (abreviado AFD) es un autómata finito que además es un sistema determinista; es decir, **para cada estado en que se encuentre el autómata, y con cualquier símbolo del alfabeto leído, existe siempre no más de una transición posible desde ese estado y con ese símbolo**.

## Definición formal

Formalmente, se define como una 5-[tupla](https://es.wikipedia.org/wiki/Tupla) (*Q*, Σ, *q0*, δ, *F*) donde:[1](https://es.wikipedia.org/wiki/Aut%C3%B3mata_finito_determinista#cite_note-hopcroft-1)​

* Q es un conjunto de [estados](https://es.wikipedia.org/wiki/Estado_(inform%C3%A1tica));
* Σ es un [alfabeto](https://es.wikipedia.org/wiki/Alfabeto);
* q 0 ∈ Q es el estado inicial;
* δ : Q × Σ → Q es una [función de transición](https://es.wikipedia.org/wiki/Funci%C3%B3n_de_transici%C3%B3n);
* F ⊆ Q es un conjunto de estados finales o de aceptación.

En un AFD no pueden darse ninguno de estos dos casos:

* Que existan dos transiciones del tipo δ(*q*,*a*)=*q*1 y δ(*q*,*a*)=*q*2, siendo *q*1 ≠ *q*2;
* 

2. **Gramáticas formales**

Gramáticas regulares

Las gramáticas formales definen un lenguaje describiendo cómo se pueden generar las cadenas del

lenguaje.

Una gramática formal es una cuadrupla G = (N, T, P, S) donde

Generan los lenguajes regulares (aquellos reconocidos por un autómata finito). Son las gramáticas

gramáticas pueden ser:

A → aB ó A → a B ∈ N

a ∈ T.

**Recursiva**

Al analizar la recursividad en gramáticas, y por lo tanto también en el lenguaje generado, podemos hacer una analogía con el concepto de función recursiva en el ámbito de la programación, donde una función es recursiva cuando se llama a sí misma.

**Producciones recursivas**

Una producción es recursiva cuando el símbolo no terminal del lado izquierdo de la regla de producción, aparece también en el lado izquierdo de la misma.

Las siguientes producciones son recursivas:   A := 0A1, B := BA01

**Producciones recursivas por izquierda**

Una producción es recursiva por izquierda cuando el símbolo no terminal del lado izquierdo de la regla de producción, aparece en primer lugar en el lado derecho de la misma.

Ejemplo:    A := A1101

### ****3.Análisis léxico****

* **Tokenización**

Un **analizador léxico** o **analizador lexicográfico** (en [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s) *scanner* o *tokenizer*) es la primera fase de un [compilador](https://es.wikipedia.org/wiki/Compilador), consistente en un [programa](https://es.wikipedia.org/wiki/Programa_(inform%C3%A1tica)) que recibe como entrada el [código fuente](https://es.wikipedia.org/wiki/C%C3%B3digo_fuente) de otro programa (secuencia de caracteres) y produce una salida compuesta de [*tokens*](https://es.wikipedia.org/wiki/Token_(programaci%C3%B3n)) (componentes léxicos) o símbolos. Estos *tokens* sirven para una posterior etapa del proceso de traducción, siendo la entrada para el [analizador sintáctico](https://es.wikipedia.org/wiki/Analizador_sint%C3%A1ctico) (en inglés *parser*).

**Expresiones regulares**

Define un conjunto de caracteres y equivale a cualquiera de

Ellos.

Si el primer carácter es un circunflejo (^), equivale a cualquier

Carácter excepto los definidos en el conjunto.

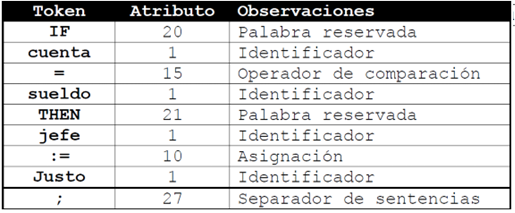
El carácter “-” se puede utilizar para definir rangos.

Los caracteres “-” y “]” pueden ser incluidos en el conjunto

**Reconocimiento de palabras clave**:

El análisis léxico de palabras es una **técnica empleada para descomponer un texto en sus elementos básicos llamados tokens**. Este proceso permite identificar palabras clave, operadores, literales e identificadores que son esenciales en la comprensión de lenguajes de programación.

**Identificación de tokens**



### ****4. Análisis sintáctico****

* **Tipos de análisis sintáctico**:

El análisis sintáctico desarrolla el esqueleto de toda la fase de

análisis

• Utiliza el analizador léxico como una rutina dentro del

análisis sintáctico ( getNextToken() )

• Integra el análisis semántico como un conjunto de rutinas a

ejecutar durante la comprobación de la sintaxis

Análisis sintáctico ascendente.

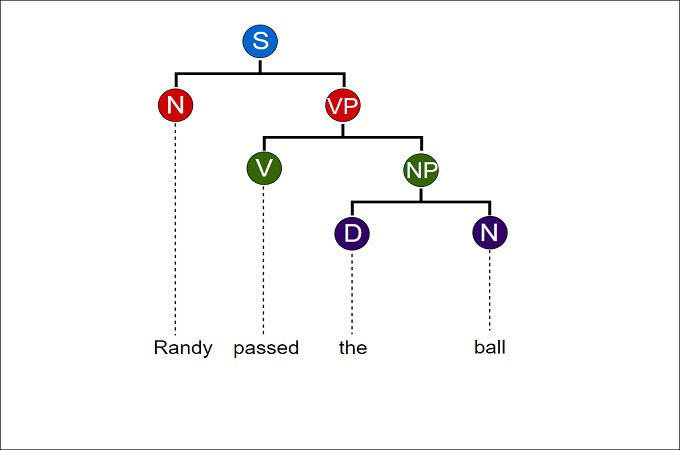
El análisis sintáctico ascendente construye la inversa de la

derivación por la derecha.

• Construye el árbol de análisis sintáctico de las hojas a la raíz.

**Árboles de análisis sintáctico**:

También conocido como árbol de derivación, este diagrama muestra toda la estructura sintáctica de una cadena, que se utiliza principalmente en la programación informática. Aparte de eso, también se puede utilizar en lingüística y a menudo se menciona como árbol sintáctico.



### ****5. Algoritmos de análisis de cadenas****

El **algoritmo de búsqueda de cadenas Boyer-Moore** es un particularmente eficiente [algoritmo de búsqueda de cadenas](https://es.wikipedia.org/wiki/Algoritmos_de_b%C3%BAsqueda_de_subcadenas), y ha sido el punto de referencia estándar para la literatura de búsqueda de cadenas práctica.[1](https://es.wikipedia.org/wiki/Algoritmo_de_b%C3%BAsqueda_de_cadenas_Boyer-Moore#cite_note-1)​ Fue desarrollado por [Bob Boyer](https://es.wikipedia.org/w/index.php?title=Bob_Boyer&action=edit&redlink=1) y [J Strother Moore](https://es.wikipedia.org/w/index.php?title=J_Strother_Moore&action=edit&redlink=1) en 1977. El [algoritmo](https://es.wikipedia.org/wiki/Algoritmo) [preprocesa](https://es.wikipedia.org/wiki/Preprocesador" \o "Preprocesador) la [cadena](https://es.wikipedia.org/wiki/Cadena_de_caracteres) objetivo (clave) *que* está siendo buscada, pero no *en* la cadena en que se busca (no como algunos algoritmos que procesan la cadena en que se busca y pueden entonces [amortizar](https://es.wikipedia.org/wiki/Amortizaci%C3%B3n) el coste del preprocesamiento mediante búsqueda repetida). El [tiempo de ejecución](https://es.wikipedia.org/wiki/Tiempo_de_ejecuci%C3%B3n) del algoritmo Boyer-Moore, aunque es lineal en el tamaño de la cadena siendo buscada, puede tener un factor significativamente más bajo que muchos otros algoritmos de búsqueda: no necesita comprobar cada carácter de la cadena que es buscada, puesto que salta algunos de ellos. Generalmente el algoritmo es más rápido cuanto más grande es la clave que es buscada, usa la información conseguida desde un intento para descartar tantas posiciones del texto como sean posibles en donde la cadena no coincida.

## Cómo funciona el algoritmo

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - | - | - | - | - | - | - | X | - | - | - | - | - | - | - |
| A | N | P | A | N | M | A | N | - | - | - | - | - | - | - |
| - | A | N | P | A | N | M | A | N | - | - | - | - | - | - |
| - | - | A | N | P | A | N | M | A | N | - | - | - | - | - |
| - | - | - | A | N | P | A | N | M | A | N | - | - | - | - |
| - | - | - | - | A | N | P | A | N | M | A | N | - | - | - |
| - | - | - | - | - | A | N | P | A | N | M | A | N | - | - |
| - | - | - | - | - | - | A | N | P | A | N | M | A | N | - |
| - | - | - | - | - | - | - | A | N | P | A | N | M | A | N |

La X en la posición 8 excluye todas la 8 posibles posiciones de comienzo mostradas.

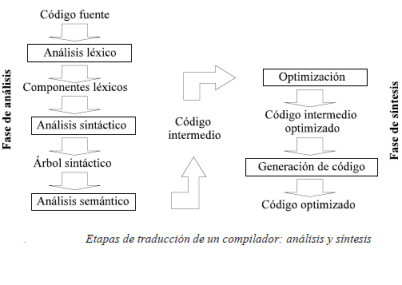
A la gente frecuentemente le sorprende el algoritmo de Boyer-Moore, cuando lo conoce, porque en su verificación intenta comprobar si hay una coincidencia en una posición particular marchando hacia atrás. Comienza una búsqueda al principio de un texto para la palabra "[ANPANMAN](https://es.wikipedia.org/wiki/Anpanman)", por ejemplo, comprueba que la posición octava del texto en proceso contenga una "N". Si encuentra la "N", se mueve a la séptima posición para ver si contiene la última "A" de la palabra, y así sucesivamente hasta que comprueba la primera posición del texto para una "A".

### ****6. Compiladores e intérpretes****

# **Estructura de un compilador**

Un compilador identifica los significados de las diferentes construcciones presentes en la definición del propio lenguaje.

En un compilador pueden distinguirse dos fases principales: una fase de **análisis**, en la que la estructura y el significado del código fuente se analiza; y otra fase de **síntesis**, en la que se genera el programa objeto.

Otra forma alternativa de estructurar las fases de las que se compone un compilador  
es distinguiendo entre: front-end, parte en la que se analiza el código, se comprueba  
su validez, se genera el árbol de derivación y se rellena la tabla de símbolos; y backend,  
donde se genera el código máquina.

**Fases del compilador**:

* **Lenguajes de programación utilizados en la creación de compiladores**: como **Lex y Yacc**.
* C
* C++
* Java
* Rust

### ****7. Aplicaciones prácticas de autómatas****

### Reconocimiento de patrones en diferentes áreas.

### El reconocimiento de patrones es una parte fundamental de la inteligencia artificial y se utiliza en una amplia variedad de aplicaciones. En este artículo, exploraremos los tres componentes principales del reconocimiento de patrones y proporcionaremos ejemplos de su aplicación en diferentes campos.

**Procesamiento de lenguaje natural** y su uso en inteligencia artificial.

utiliza en el filtrado del correo electrónico, los asistentes de voz, los [agentes de IA](https://botpress.com/es/blog/what-is-an-ai-agent), los motores de búsqueda, la predicción de texto y [los chatbots de IA](https://botpress.com/es/blog/an-introduction-to-ai-chatbots). Pero aunque está en todas partes, la PNL es un proceso complejo que pocos tenemos en cuenta cuando utilizamos un servicio de traducción de idiomas o nuestro teléfono predice la siguiente palabra que vamos a escribir.

El primer paso para entender la PNL es definir qué es. Empecemos.

## ¿Qué es el procesamiento del lenguaje natural?

El procesamiento del lenguaje natural (PLN) es una rama de la inteligencia artificial que se centra en la interacción entre ordenadores y humanos a través del lenguaje natural. Permite a las máquinas comprender, interpretar y generar el lenguaje humano de forma significativa y útil.

### **Análisis de texto y minería de datos**

El texto es un requisito básico en nuestra vida. Toda la información, los detalles y las interpretaciones se realizan enviando mensajes de texto y decodificando el texto. El texto que usamos en nuestra vida digital diaria es estándar, y hay algunos textos que solo usan las autoridades superiores que están encriptados.

Estos textos se extraen cuidadosamente y también hay datos para autoridades superiores, como la inteligencia artificial.

### ****8. Lenguaje de programación C y C++****

### Qué es una librería en C

Las librerías en el lenguaje C son fundamentales para el desarrollo de software eficiente y modular.

Entender su concepto básico, importancia y variedad es esencial para cualquier programador que desee aprovechar al máximo esta herramienta poderosa.

### Concepto básico de librerías en C

**Las librerías en C son archivos que contienen implementaciones de funciones y procedimientos que pueden ser utilizados en diferentes programas.** Este enfoque modular permite a los programadores dividir su código en unidades más manejables y facilita la reutilización de funciones en varios proyectos.

**Imagina las librerías en C como cajas de herramientas llenas de funciones listas para ser utilizadas en cualquier momento.** A medida que exploramos este concepto, descubrirás cómo las librerías en C ofrecen flexibilidad y modularidad en el desarrollo de software.

### Importancia de las librerías en C

**La importancia de las librerías en C radica en su capacidad para promover la reutilización del código y simplificar el desarrollo de software.** Al encapsular funciones específicas en archivos separados, los programadores pueden crear programas más limpios y fáciles de mantener.

## Cuál es la función de una biblioteca en C++Las bibliotecas en C++ son auténticos tesoros para los programadores, ya que contienen un conjunto de funciones y procedimientos que simplifican tareas comunes.Su función principal radica en proporcionar una amplia gama de herramientas predefinidas que agilizan el desarrollo de programas.

Imagina las bibliotecas como una biblioteca física con estantes llenos de libros, pero en este caso, cada libro es una función que puedes utilizar en tu código.

