PARADIGMAS 4 (1) 81-92 (2012)
Revista del Departamento Académico de Ingeniería.
Universidad Ricardo Palma. ISSN: 2220-2056

Procesamiento probabilístico del lenguaje

Probabilistic processing language

Augusto Cortez Vásquez[1]

Resumen

El presente articulo describe el procesamiento del lenguaje desde el enfoque probabilístico. Utiliza las gramáticas libres de contexto en la que se asocia una probabilidad a cada una de las reglas de derivación de la gramática que serán utilizadas durante la fase de análisis sintáctico de un lenguaje. El procesamiento probabilístico del lenguaje ha sido posible dado a los grandes volúmenes de información que se dispone en los denominados corpus. Un modelo probabilístico del lenguaje define una distribución de la probabilidad sobre un conjunto de cadenas. Algunos ejemplos de modelos: el bigram y el trigram, son modelos del lenguaje usados en el reconocimiento del habla. Un modelo unigram asigna una probabilidad P (W) a cada palabra del léxico. El modelo asume que las palabras están elegidas independientemente, así que la probabilidad de una secuencia es el producto de la probabilidad de sus palabras.

Palabras claves

Lenguaje, procesamiento de lenguaje, procesamiento probabilístico del lenguaje, lenguajes libre de contexto, análisis sintáctico probabilístico.

Abstract

This article describes the processing of language from the probabilistic approach. Using the context-free grammars in the that is associated with a probability to each of the derivation rules of grammar that will be used during the analysis phase of a syntactic language. The probabilistic processing of language has been possible because of the large volumes of data that is available in the so-called corpus. A probabilistic model of the language defines a probability distribution on a set of strings. Some examples of models: the bigram and trigram, are models of the language used in speech recognition. A design at Unigram model assigns a probability of P (W) to each word in the lexicon. The model assumes that the words are chosen independently, so the probability of a sequence is the product of the probability of their words.

Key words

Language, language processing, probabilistic processing language, context-free languages, probabilistic parsing.

1 Licenciado en Computación. Magíster en Computación e Informática. Docente URP

1. Introducción

Un agente puede comunicarse con otro agente (ser humano o software), a través de un lenguaje común. El lenguaje esta compuesto de expresiones, que requieren ser analizadas para extraer su significado, y esto es posible dado que dichas expresiones contienen palabras cortas y restringidas en un dominio limitado. El proceso de análisis consta de análisis lexicográfico, análisis sintáctico y análisis semántico.

Durante toda la historia de humanidad el conocimiento, en su mayor parte se comunica, se guarda y se maneja en la forma de lenguaje natural –griego, latín, inglés, español, etc. La época actual no es ninguna excepción: el conocimiento sigue existiendo y creándose en la forma de documentos, libros, artículos, aunque éstos se guardan en forma electrónica, o sea digital. El gran avance es que en esta forma, las computadoras ya pueden ser una ayuda enorme en el procesamiento de este conocimiento. Sin embargo, lo que es conocimiento para nosotros –los seres humanos– no lo es para las computadoras. Son los archivos, unas secuencias de caracteres, y nada más. Una computadora puede copiar tal archivo, respaldarlo, transmitirlo, borrarlo –como un burócrata que pasa los papeles a otro burócrata sin leerlos. Pero no puede buscar las respuestas a las preguntas en este texto, hacer las inferencias lógicas sobre su contenido, generalizar y resumirlo –es decir, hacer todo lo que las personas normalmente hacemos con el texto. Dado que no las puede entender[1,17].

Para combatir esta situación, se dedica mucho esfuerzo, sobre todo en los países más desarrollados del mundo, al desarrollo de la ciencia que se encarga de habilitar a las computadoras a entender el texto. Esta ciencia, dependiendo del enfoque que se le de recibe varios nombres: procesamiento de lenguaje natural, procesamiento de texto, tecnologías de lenguaje, lingüística computacional. En todo caso, se trata de procesar el texto por su sentido y no como un archivo binario. La técnica que presentaremos en este articulo es de un enfoque probabilístico

2. Marco conceptual

Lenguaje

Un lenguaje es un conjunto de frases definidas por un conjunto de reglas... Estas reglas se pueden expresar de muchas formas. Cuando esto ocurre se dice que el lenguaje esta descrito por comprensión. Otra forma de describir el lenguaje es describiendo uno a uno las frases que la componen, en este caso estamos frente a una definición del lenguaje por extensión. En el contexto de definición por comprensión, existen varias formas de describir un lenguaje: descripción algebraica, por expresión regular, en forma recursiva etc[14].

Descripción gramatical

Sea G una gramática, definimos el lenguaje definido por una gramática y la denotamos L(G) de la siguiente forma:

$$L(G)=\{w / S \longrightarrow {}^{*}w, \text{ y además } w \in Vt^{*}\},$$

L contiene secuencias de terminales que son generadas a partir del axioma de G.
$L=\{a^{m} b, m \geq 0\}$ descripción algebraica

Podemos hallar G tal que L=L(G)

$G(V_N, V_T, S, P)$ donde

$V_N = \{S\}$

$V_T = \{a, b\}$

$P: \{S \longrightarrow aS/b\}$

$S \longrightarrow b$

$S \longrightarrow aS \longrightarrow ab$

$S \longrightarrow aS \longrightarrow aaS \longrightarrow aab$

$S \longrightarrow aS \longrightarrow aaS \longrightarrow aaaS \longrightarrow aaab$

luego b,ab, aab, aaab $\in$ L(G).

Sintaxis de un lenguaje

En esta sección se define la sintaxis como el conjunto de reglas de formación de las oraciones de un lenguaje. Una oración es válida si es reconocida, es decir, cumple con las reglas de sintaxis. En un lenguaje natural, como el español, la sintaxis de una oración es determinada por la secuencia:

sujeto + verbo + predicado

Así las sentencias «Juan estudia mucho», «María Esther estudia poco» son reconocidas la disposición de palabras en una oración para mostrar su relación. Describe la secuencia de símbolos que constituyen programas válidos.

En un lenguaje formal (de programación):

la frase **a = b + c** representa una secuencia válida de símbolos,
pero **c = b a +** no lo es.

Esto se justifica dado que en una sentencia de asignación el lado izquierdo del operador de asignación debe ser un identificador y en el lado derecho debe haber una expresión aritmética válida.

La sintaxis suministra información significativa que se necesita para entender un programa y proporciona información imprescindible para la traducción del programa fuente a un programa objeto [10].

La sintaxis muestra al programador la forma como debe escribir buenos programas.
La sintaxis es más útil al usuario del lenguaje de programación que al constructor del compilador. El constructor tiene que describir la sintaxis formalmente mediante un modelo matemático-lingüístico llamado gramática de lenguaje. Podemos afirmar que la gramática es la sintaxis descrita formalmente. Cuando una persona quiere aprender un lenguaje, necesita conocer la sintaxis del lenguaje, y la puede encontrar en cualquier manual del lenguaje. Sin embargo, la gramática no le será muy útil, porque se requiere conocimientos formales de gramática para su entendimiento.

Gramática

Una gramática es un instrumento que se utiliza para especificar la sintaxis de un lenguaje. La gramática de un lenguaje describe las reglas para verificar el orden estructural de las frases que pertenecen al lenguaje. La gramática no describe la semántica o significado de las distintas proposiciones[7].

Definición formal de gramática

Una gramática se define formalmente de siguiente forma:

$G=(V_T, V_N, P, S)$ donde:

V_T : conjunto finito de símbolos terminales del lenguaje
V_N : conjunto finito de símbolos no terminales
P : conjunto finito de reglas de producción
S : Símbolo distinguido o axioma inicial a partir del cual se reconocerán las secuencias de L aplicando sucesivamente las reglas de producción.

Ejemplo

$L(G1)=\{a^m b, m > 0\}=\{b, ab, aab, aaab,...\}$

$G1(V_N, V_T, S, P)$ donde

$V_N = \{S\}$
$V_T = \{a, b\}$
P: $\{S \longrightarrow aS/b\}$

$S \longrightarrow b$
$S \longrightarrow aS \longrightarrow ab$
$S \longrightarrow aS \longrightarrow aaS \longrightarrow aab$
$S \longrightarrow aS \longrightarrow aaS \longrightarrow aaaS \longrightarrow aaab$

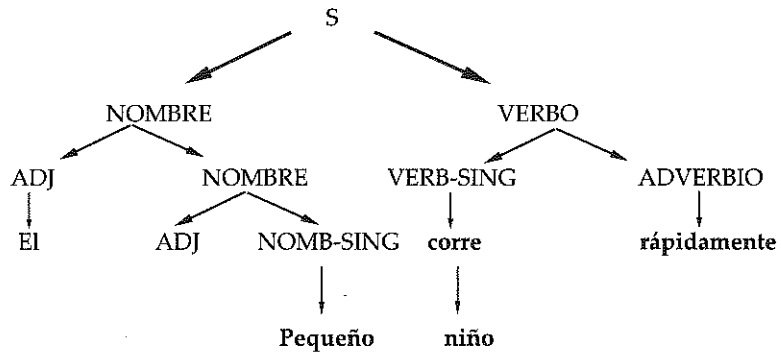
luego b,ab, aab, aaab $\in L(G_1)$.

Ejemplo

Consideremos la siguiente gramática

$G(V_N, V_T, S, P)$ donde
V_N = {S,NOMBRE,VERBO,ADJ, NOMB-SING, VERBO-SING,ADVERBIO}
V_T {El, La, Los, Las, Pequeño, traviesa, niño, niña, estudia, corre, juega, salta}

P: {
S ⟶ NOMBRE VERBO
NOMBRE ⟶ ADJ NOMBRE
NOMBRE ⟶ ADJ NOMB-SING
VERBO ⟶ VERB-SING ADVERBIO
ADJ ⟶ El /La /Los /Las
ADJ ⟶ Pequeño /traviesa
NOMB-SING ⟶ niño / niña
VERB-SING ⟶ estudia / corre / juega /salta
ADVERVIO ⟶ rápidamente / despacio / mucho
}



luego w='El Pequeño niño corre rápidamente' $\in L(G_{31})$

Traductor

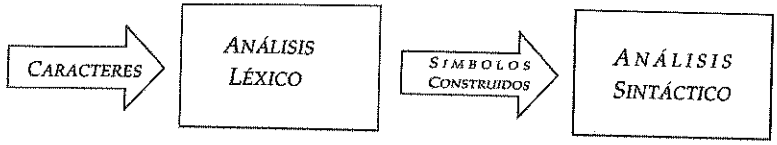
Un traductor es una máquina que tiene como entrada un texto escrito en un lenguaje L_1 y como salida un texto escrito en un lenguaje L_2. Habitualmente se denomina a L_1 lenguaje fuente y a L_2, lenguaje objeto. Los traductores son programas formales que traducen programas escritos en un lenguaje de programación independiente de la máquina en otro lenguaje.

$W_1 \in L(G_1)$ y $W_2 \in L(G_2)$

Una proposición o sentencia está conformada por una cadena de caracteres, sin embargo, por razones de análisis el compilador agrupa los caracteres en secuencias llamadas lexemas *(tokens)* que corresponden a los componentes léxicos definidos para el lenguaje. Estos componentes pueden suponerse como los axiomas o fundamentos del lenguaje, pues vienen a ser los átomos o ladrillos con lo que se construyen las frases de un lenguaje. Un componente léxico puede ser una palabra clave, un nombre de variable (identificador), una constante numérica entera o real, un operador aritmético, etc. La tarea de examinar la proposición con la finalidad de reconocer y clasificar los distintos componentes léxicos se conoce como «análisis léxico» y la parte del compilador que realiza esa función analítica se denomina «analizador léxico» *(scanner)*. Después de analizar los componentes léxicos, cada proposición del programa debe reconocerse como una frase del lenguaje, como una declaración, proposición de asignación, etc. Es decir, debe haber un orden prescrito en los componentes léxicos que conforman la proposición. Este proceso se denomina «análisis sintáctico» *(parsing)*, lo realiza la parte del compilador denominada «analizador sintáctico» *(parser)*. El último paso en el proceso básico de traducción es la generación de código objeto.

El analizador léxico lee caracteres de un alfabeto y los agrupa en símbolos (componentes léxicos) que son pasados al analizador sintáctico para la construcción de las proposiciones. Cada componente léxico o símbolo construido por el lexicográfico corresponde a una variable terminal de la gramática del lenguaje que analizará el analizador sintáctico.

Los componentes léxicos se consideran como los fundamentos del lenguaje. La tarea de examinar el programa fuente con el fin de reconocer y clasificar los distintos componentes léxicos se conoce como «análisis léxico» y la parte del compilador que realiza esta función se denomina «analizador léxico» (Beck 1988).



En forma general cualquier sistema automático de procesamiento de lenguaje utiliza las estructuras del lenguaje, para generar los procesos de análisis que se muestran en la siguiente tabla:

Estructura del lenguaje	Proceso de análisis
Fonética y morfología: estudia los sonidos para formar palabras a partir de las unidades pequeñas	**Morfológico-lexico:** elabora una cadena de caracteres a partir de los sonidos y la transforma en unidades significativas (unidades léxicas) utilizando el diccionario y las reglas morfológicas
Sintaxis: estudia las combinaciones de las palabras para formar frases correctas	**Sintáctico:** a partir de las unidades léxicas se genera una estructura representativa (árbol, red etc.)
Semántica: estudia el significado de las palabras y su forma de combinación para llegar al significado de una oración	**Semántico:** a partir del proceso sintáctico se obtiene una nueva estructura o forma lógica que representa el sentido de la oración
Gramática de contexto: estudia como el contexto afecta a la interpretación de las oraciones	**Contextual:** utiliza la estructura semántica para obtener la interpretación final de la oración según el contexto.

Funciones características de un analizador léxico

El análisis léxico representa el nivel mas bajo dentro del procesamiento de lenguaje, ya que a partir del resultado obtenido se construyen las palabras que mas tarde formaran las oraciones tendiendo en cuenta los requisitos morfológicos, sintácticos y semánticos necesarios para ello [Pajares]. El analizador lexicográfico explora la cadena de entrada y verifica que cada lexema corresponda a un componente léxico definido. Si existe un lexema que no corresponde a ningún componente léxico, se dice que ha ocurrido un error lexicográfico. Alternativamente durante el proceso de análisis lexicográfico, se eliminan los espacios en blanco, así como se puede generar un listado para el compilador.

Para realizar el análisis lexicográfico deberá primero definirse cuales son los componentes léxicos válidos. Luego cuando se realice el análisis lexicográfico se explorará cada lexema, este será válido si corresponde a algún componente léxico, en otro caso se dirá que ocurrió un error.

Funciones características de un analizador sintáctico

El analizador sintáctico recibe una secuencia de tokens del analizador léxico y decide si la secuencia esta bien escrita o no. Para ello hace uso de las reglas de derivación de la gramática del lenguaje.
El proceso de análisis sintáctico puede describirse como el intento de construir un árbol de análisis

sintáctico para la frase que se quiere reconocer. Aunque en la practica los árboles sintácticos no se implementan porque ocuparían mucho espacio. Por esta razón se usan como alternativa los autómatas de pila, que guarda información en la pila acerca de los nodos del árbol sintáctico relevantes en cada fase del proceso de análisis.

La sintaxis de las construcciones de los lenguajes de programación pueden describirse por medios de gramáticas libres del contexto[Aho y Ullman, 1990]

Ventajas

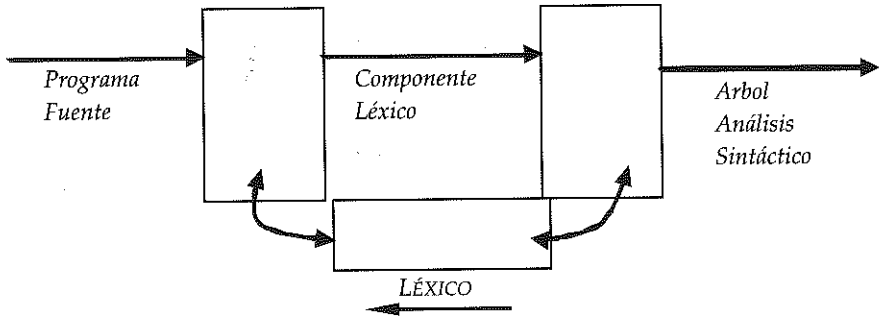
1. La gramática proporciona una especificación precisa, fácil de entender de un lenguaje de programación

2. A partir de algunas clases de gramáticas se puede construir automáticamente un analizador eficiente que determine si un programa fuente esta sintácticamente bien formado.

3. Los lenguajes evolucionan con el tiempo, adquiriendo nuevos construcciones y realizando tareas adicionales. Estas nuevas construcciones se pueden añadir con mas facilidad a un lenguaje cuando existe una aplicación basada en una descripción gramatical.

4. Una gramática imparte una estructura a un lenguaje de programación útil para la traducción de un programa fuente a código objeto correcto y para la detección de errores

5. El proceso de construcción del analizador sintáctico puede revelar ambigüedades sintácticas y otras construcciones difíciles de analizar que podrían pasar sin ser detectadas en la fase inicial del diseño de un lenguaje y de su compilador.



Análisis sintáctico descendente

Intenta construir un árbol de análisis sintáctico, empezando desde la raíz y descendiendo hacia las hojas. Lo que es lo mismo que intentar obtener una derivación por la izquierda para una cadena de entrada, comenzando desde la raíz y creando los nodos del árbol en orden previo

Análisis sintáctico ascendente

Intenta construir un árbol de análisis sintáctico, empezando desde las hojas (la cadena) y ascendiendo hacia la raíz. Lo que es lo mismo que intentar obtener una reducción desde una cadena hasta llegar al axioma.

Modelos probabilísticos del lenguaje MPL

Un MPL define una distribución de la probabilidad sobre un conjunto (posiblemente infinito) de cadenas. Un modelo Unigram asigna una probabilidad P(w) a cada palabra del léxico. El modelo asume que las palabras están elegidas independientemente, así que la probabilidad de una secuencia es el producto de la probabilidad de sus palabras, dada por π P(wi)

Los modelos probabilísticos tienen varias ventajas. Pueden ser entrenados convenientemente a partir de los datos; el aprendizaje es justo una cuestión de contar ocurrencias. También se consideran más robustos, ya que reflejan el hecho de que no todos los hablantes están de acuerdo en que frases son parte de un idioma actualmente, y pueden utilizarse para la desambiguación: la probabilidad de elegir la interpretación más probable.

Un modelo probabilístico del lenguaje define una distribución de la probabilidad sobre un conjunto de cadenas. Algunos ejemplos de modelos: el bigram y el trigram, son modelos del lenguaje usados en el reconocimiento del habla. Un modelo unigram asigna una probabilidad P (W) a cada palabra del léxico. El modelo asume que las palabras están elegidas independientemente, así que la probabilidad de una secuencia es el producto de la probabilidad de sus palabras, dada por:

$$\pi\, P(W_i)$$

Un modelo bigram asigna una probabilidad

$$\pi\, P(W_i/W_{i-1})$$

a cada palabra, dependiendo de la palabra anterior.
En general, un modelo n-gram condiciona una palabra a las N – 1 anteriores, asignando una probabilidad

$$\pi\, P(W_i/W_{i-(n-1)}....W_{i-1})$$

Gramáticas probabilísticas independiente del contexto GPLC

Los modelos probabilísticos n-gram de las gramáticas independientes del contexto se aprovechan de la estadística de col-ocurrencia en los corpus, pero no aportan ninguna noción de gramática para valores mayores que n. Un modelo alternativo del lenguaje es la gramática probabilística independiente del contexto GPLC. El modelo GPLC la probabilidad de una secuencia P(palabra), es justo la suma de las probabilidades de sus arboles de análisis. La probabilidad de un árbol dado es el producto de las probabilidades de todas las reglas que constituyen los nodos del árbol[17].

hP: {	S	⟶	NOMBRE VERBO	[1.0]
	NOMBRE	⟶	ADJ NOMBRE	[0.4]
	NOMBRE	⟶	ADJ NOMB-SING	[0.6]
	VERBO	⟶	VERB-SING ADVERBIO	[1.0]
	ADJ	⟶	El	[0.25]
	ADJ	⟶	/La	[0.25]
	ADJ	⟶	/Los	[0.15]
	ADJ	⟶	/Las	[0.15]
	ADJ	⟶	Pequeño/traviesa	[0.20]
	NOMB-SING	⟶	niño	[0.5]
	NOMB-SING	⟶	/niña	[0.5]
	VERB-SING	⟶	estudia	[0.27]
	VERB-SING	⟶	/corre	[0.16]
	VERB-SING	⟶	/juega	[0.34]
	VERB-SING	⟶	/salta	[0.23]
	ADVERVIO	⟶	rápidamente	[0.45]
	ADVERVIO	⟶	/despacio	[0.28]
	ADVERVIO	⟶	/mucho	[0.27]
}				

En la mayoría de los idiomas modernos existen palabras que se repiten en un dado texto. Resulta interesante realizar, para un dado texto, una estadística de estas palabras que se repiten, ordenándolas de acuerdo a su frecuencia de aparición y calculando su probabilidad de aparición. Esta estadística de las palabras que se repiten se llama unigrama (unigram en inglés). El objeto de esta actividad es estudiar el histograma de aparición de las palabras o más precisamente la distribución de probabilidades que siguen los unigramas. Para tal fin se propone tomar un texto, con suficientes palabras (más de 3000) e identificar cuantas veces las palabras se repiten[Bradley, 2009].

Existen algunos software como el HWFC (Hermetic Word Frequency Counte) que scanean un texto o un archivo y cuenta el numero de ocurrencias de las diferentes palabras que están contenidas, mostrándolas alfabéticamente con sus respectivas frecuencias.[17].

Aprendizaje de probabilidades para GPLC

Para crear una GPLC, debemos primero crear la gramática LC, y luego fijamos las probabilidades de cada regla . Esto sugiere que aprender la gramática a partir de los datos seria mejor que una aproximación con la ingeniería de conocimiento. Esto es de por si complejo, sin embargo podríamos considerar datos analizados en arboles por lingüistas, la creación de este corpus demanda una gran inversión

Aprendizaje de la estructura de las reglas para GPLC

Si no conocemos la estructura de las reglas gramaticales, nos enfrentamos al problema de no saber cuantas reglas existen y por cada no terminal cuantas reglas alternativas tenemos. Esta tarea se puede simplificar si normalizamos la gramática en la forma normal de Chomsky, lo cual reduciría las alternativas por cada no terminal.

Forma normal de Chomsky

Sea L lenguaje libre del contexto donde $\lambda \in L$, $\exists$ G libre del contexto tal que L(G) = L, en donde el lado derecho de cualquier regla de producción consiste de un solo terminal o exactamente dos no terminales.

La gramática que cumple estas características se dice que está en la forma normal de Chomsky (FNC), en honor a Noam Chomsky[1].

La siguiente gramática está en la FNC

A	$\longrightarrow$	BC	$A, B, C \in V_N$	Forma normal
A	$\longrightarrow$	b	$b \in V_T$	de Chomsky

3. Discusión

Tras considerar que en un lenguaje existe un conjunto inmenso de palabras, y que cada una aparece con una frecuencia diferente en un texto, es razonable asignarle a cada regla gramatical la probabilidad asociada a su frecuencia de aparición. El modelo n-gram es uno de los modelos estadísticos del lenguaje más simples al mismo tiempo que uno de los más útiles. Se basa en el supuesto de que sólo las n-1 palabras anteriores tienen efecto sobre las probabilidades de la siguiente palabra.

Una objeción común a las gramáticas en general es que los corpus contienen frecuentemente estructuras no gramaticales que las gramáticas estándar rechazan lo que se conoce como agramaticalidad. No obstante, una solución sería simplemente incluir reglas que generen todas las cosas que pueden aparecer en un texto español, gramaticales o no, estas gramáticas fallarían en distinguir entre cadenas anómalas que aparecen rara vez y aquellas que son verdaderamente parte del lenguaje. Digamos por tanto que es conveniente que las GPLC pueden asignar probabilidades altas a las sentencias que habitualmente consideramos gramaticales y otras mucho más bajas a los casos no gramaticales.

CONCLUSIONES

El procesamiento probabilístico del lenguaje constituye una técnica que ha sido posible a partir de la disponibilidad de grandes bases de datos de lenguaje natural lo que ha dado nacimiento a los corpus. La construcción de corpus puede hacerse manualmente, pero claramente es más sencillo usar un programa para hacer esta operación. En Internet existen muchos programas que pueden hacer este análisis. Uno de ellos es Hermetic Word Frequency Counter. La conclusión importante es que los corpus determinan la probabilidad asociada a cada regla gramatical del lenguaje en cuestión, pero que sin embargo es menester tener en cuenta las estructuras no gramaticales

REFERENCIAS BIBLIOGRÁFICA

[1] [AHO 1990] AHO A.,SETHI,ULLMAN "Compiladores, principios, técnicas y herramientas"; Addison-Wesley1990, Wilmington-Delaware EUA.

[2] [BECK 1988] BBECK LELAND "Software de Sistemas"
Addisson Wesley iberoamericana Wilmington Delaware 1988

[3] [Bradley, 2009] J. R. BRADLEY AND D. L.FARNSWORTH, "What is Benford's law?", Teaching Statistics. Vol. 31,(1), 2-5 (2009)

[4] [BROOKSHEAR 1993] BROOKSHEAR J. GLEAN
Teoría de la computación
Addisson Wesley iberoamericana Wilmington Delaware 1993

[5] [CORTEZ 1999] CORTEZ VÁSQUEZ, AUGUSTO. " Matemáticas Discretas", UNMSM EAPIS 1999.

[6] [CORTEZ 2002] CORTEZ VÁSQUEZ, AUGUSTO. " Lenguajes y compiladores", UNMSM Lima 2002.

[7] [CORTEZ 2011] CORTEZ VÁSQUEZ, AUGUSTO. " Traductores", UCSS Lima 2011.

[8] [DEITEL 1987] DEITEL HARVEY M.. "Introducción a los sistemas operativos"; Addison-Wesley, Iberoamenricana 1987 Wilmigton Deleware E.U.A:

[9] [HOPCROFT 1993] HOPCROFT JHON, ULLMAN JEFFREY
Introducción a la teoría de autómatas Edit. CECSA 1993

[10] [JOHNSONBAUGH 1999] JOHNSONBAUGH RICHARD "Mateamticas Discretas"; Prentice Hall 1999

[11] [KOLMAN 1997] Kolman-Busby-Ross "Matematicas Discretas"; Prentice Hall 1997

[12] [PEñA 1998] RICARDO PEÑA MARI "Diseño de programas, Formalismo y abstracción"
Prentice Hall Madrid 1998

[13] [PRATT 1988] TERRENCE W. PRATT. "Lenguajes de programación, Diseño e implementación"; Prentice Hall Hispanoamericana 1988

[14] [RUSELL 2008] STUART RUSSELL. "Inteligencia artificial, un enfoque moderno"; Prentice Hall Hispanoamericana 2008

[15] [SETHI 1992] SETHI, RAVI "LENGUAJES DE PROGRAMACION, Conceptos y Constructores"; Addison-Wesley, 1992 .

[16] [TEUFEL 1990] TEUFEL-SMITHD-TEUFEL. "Compiladores, Conceptos fundamentales"; Addison-Wesley, 1990.

[17] http://eprints.ucm.es/tesis/19911996/X/2/X2002401.pdf

[18] http://sinai.ujaen.es/sepln/ojs/ojs-2.3.5/index.php/pln/article/view/4212/2545
Sociedad española para el procesamiento del lenguaje natural