UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

Seminario de análisis del discurso

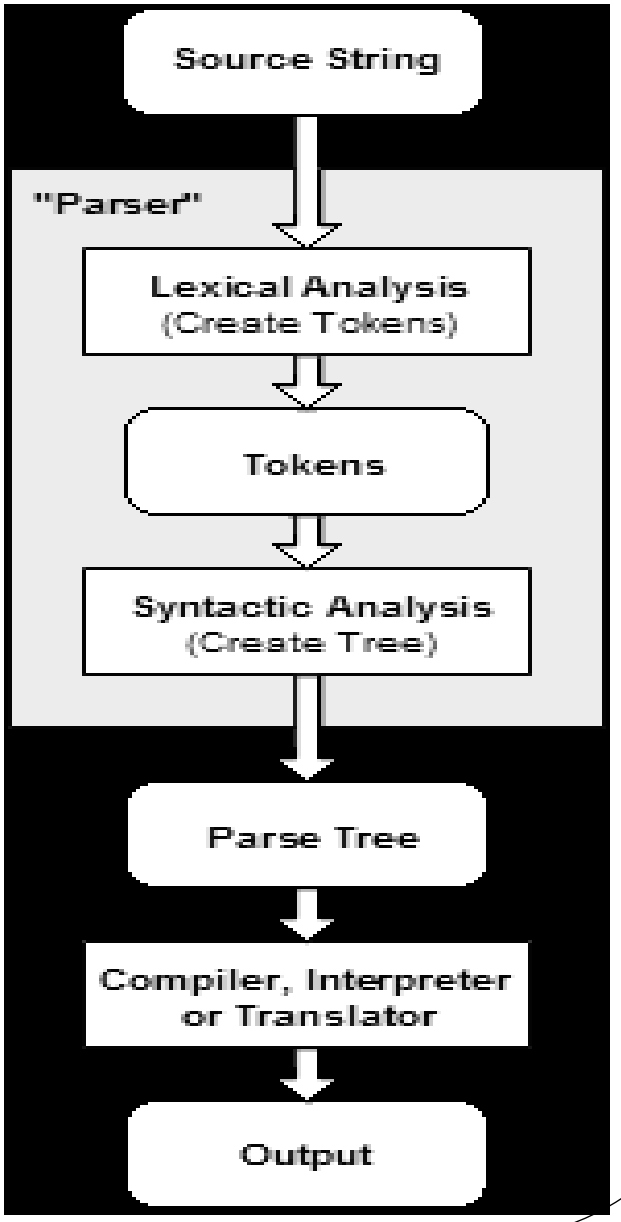
Dr. César Antonio Aguilar
Facultad de Lenguas y Letras
27/09/2010

CAguilar@iingen.unam.mx

Análisis sintáctico en corpus (1)

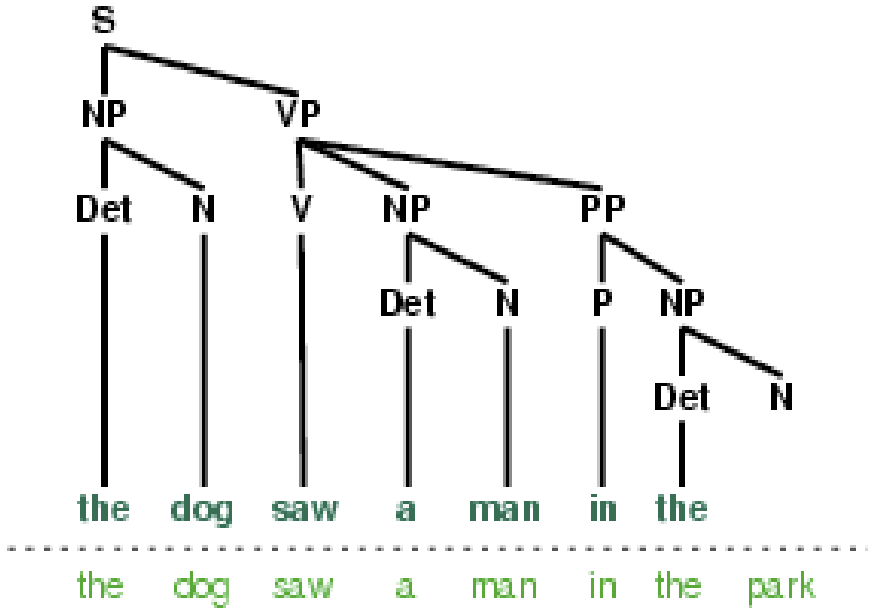
En esta clase, terminaremos de revisar lo que puede hacerse usando NLTK para el análisis de corpus lingüísticos. En este caso, nos enfocaremos en el análisis morfosintáctico.

Como veremos, NLTK nos ofrece varias ventajas para hacer esta clase de análisis, incluyendo la posibilidad de trabajos con distintos modelos, lo que permite hacer contrastes pertinentes entre los resultados obtenidos.



Análisis sintáctico en corpus (2)

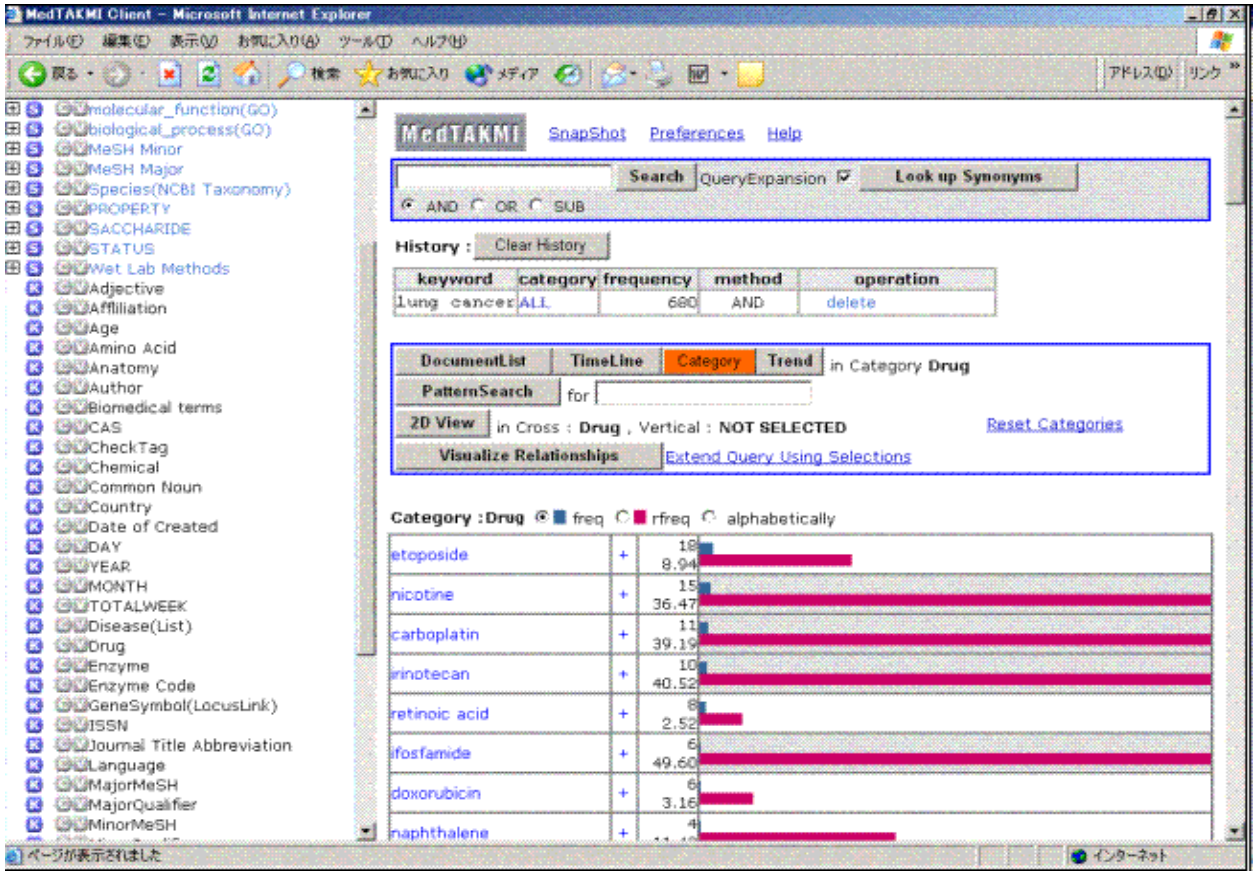
Para entender cómo se hace esta clase de análisis, requerimos entender primero algunos conceptos básicos. Veamos:



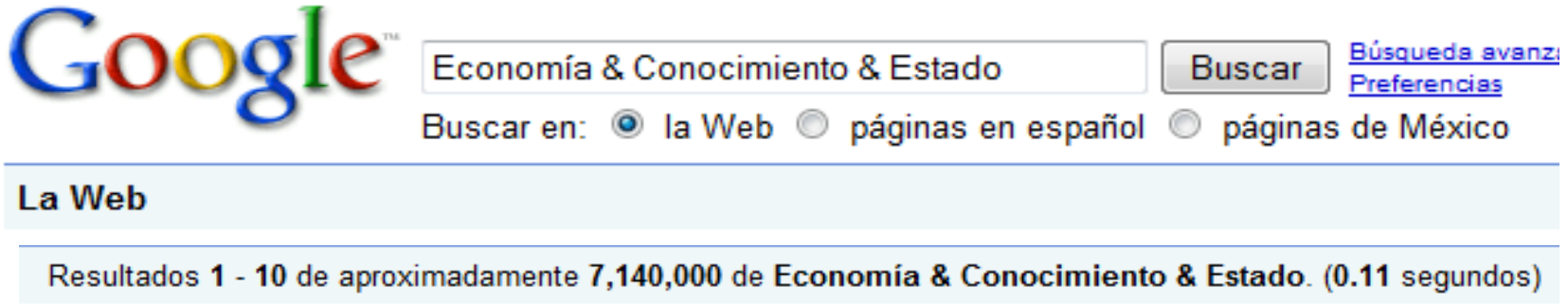
Para hacer búsquedas eficientes de combinaciones de palabras (esto es, de frases y oraciones), además de usar métodos como el reconocimiento de concordancias y colocaciones, también podemos emplear expresiones regulares junto con chunkers y parsers.

Expresiones regulares (1)

Una expresión regular es un tipo de patrón o estructura que describe un conjunto de cadenas sin enumerar sus elementos. Es “regular porque su codificación tiende a ser repetitiva (de hecho, se pueden considerar una especie de “rutina” para comunicar cierta información).



Expresiones regulares (2)



El uso de expresiones regulares es algo muy común cuando buscamos, por ejemplo, información en Internet:

Economía del **Conocimiento** | Directorio del **Estado**
No sabemos muy bien que entenderá el Sr. Presidente de España por **Economía** del **Conocimiento**, pero esperamos comprenda que la base de todo **conocimiento** es la ...
www.gobiernoelectronico.org/node/5198 - 50k - En caché - Páginas similares

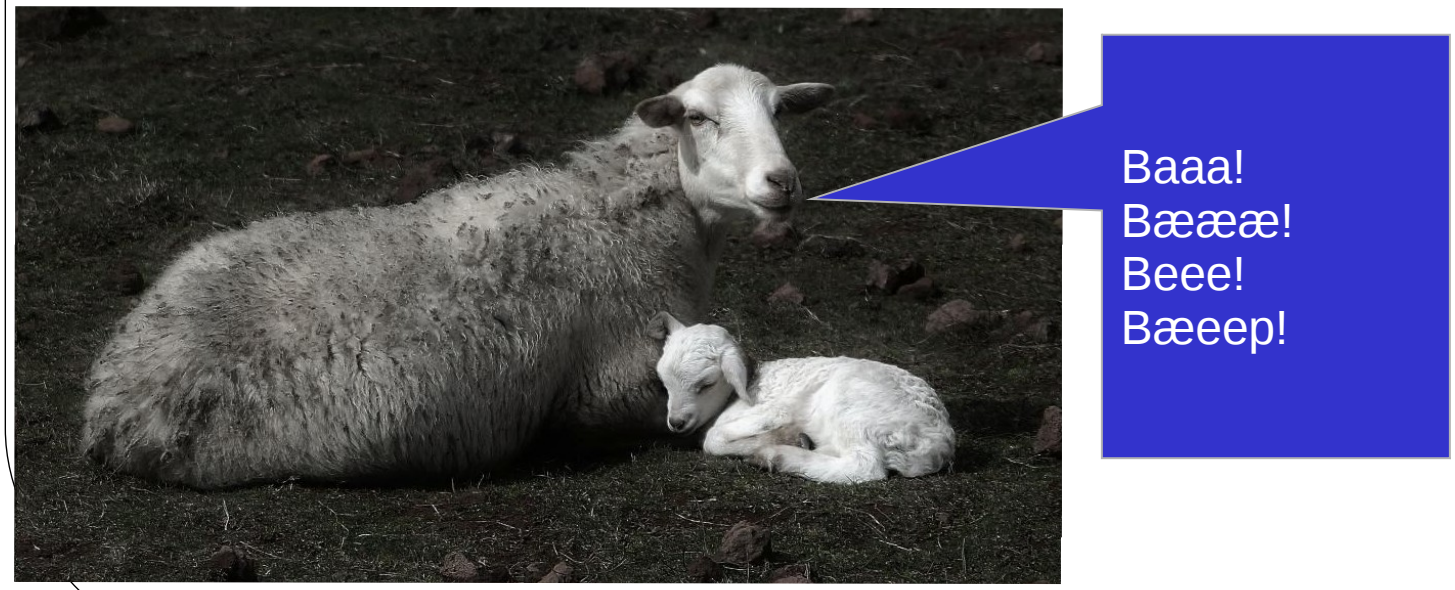
UNAM-Instituto de Investigaciones Económicas-Unidades de Investigación
Diagnóstico integral sobre el desarrollo de la **economía** del **conocimiento** en México • Alianzas estratégicas del **Estado** Mexicano con el sector empresario ...
www.iiec.unam.mx/unidades/unidad_investigacion_**conocimiento**_desarrollo.htm - 47k - En caché - Páginas similares

[PDF] 40. Manifestaciones de la **Economía** del **Conocimiento** en América
Formato de archivo: PDF/Adobe Acrobat - Versión en HTML
13 Oct 2008 ... De esta forma, hablar de **economía** del **conocimiento**, es hablar también del debate de la gobernanza en donde el **Estado** se ...
energia.guanajuato.gob.mx/gaceta/Gacetaideas/Archivos/40012008_NOTA_EDITORIAL.pdf - Páginas similares

Expresiones regulares (3)

Dan Jurafsky (Universidad de Stanford) y James Martin Universidad de Colorado) plantean el siguiente ejemplo. Pensemos que las ovejas tienen un lenguaje bien definido para comunicarse. Este lenguaje es un enorme repertorio de balidos con algunas *variantes dialectales*, p. e.:

La idea de que se traten de variantes nos permite pensar que, si bien hay cambios en un plano léxico, en los planos sintácticos y semánticos siguen siendo construcciones regulares.



Expresiones regulares (4)

Podemos realizar algunas operaciones sencillas, p. e., si hacemos un análisis de balidos, podemos tomar como equivalentes todas las variantes posibles.

Expresión regular	Significado
/[bB]aaa!/	baaa! *or* Baaa!
/[aæe]/	baaa!, bæææ! *or* beee!

Expresiones regulares (5)

Otras operaciones son establecer rangos en conjuntos:

Expresión regular	Significado
/[A-Z]/	“Busca todos los caracteres que sean mayúsculas”
/[a-z]/	“Busca todos los caracteres que sean minúsculas”
/[0-9]/	“Busca todos los números de 0 a 9”

O hacer negaciones:

Expresión regular	Significado
[^A-Z]	“Busca todos los caracteres menos mayúsculas”
[^S-s]	“Busca cualquier carácter menos ‘S’ o ‘s’”
[^\.]	“No es un párrafo”
[e^]	“Cualquiera que sea ‘e’ o ‘vacío’”
a ^ b	“Busca el patrón ‘a_b’”

Expresiones regulares (6)

Finalmente, algunos caracteres opcionales:

Expresión regular	Significado
/ ? (Disyunción)/	“’Computadora’ OR ‘Ordenador’”
* (Estrella de Klenne)	“’Busca desde 0 hasta infinito’”
+ (Suma o adición)	“Busca desde 1 hasta infinito”
/ Constan**.** / (Comodín)	“’Constante OR ‘Constancia’”
/ ^ [A-Z] /	“’El Colegio de México’ OR ‘Colegio de México’”
/ ^ [^A-Z] /	“’¿Verdad?’ OR ‘Really?’”
/ \. $/	“’Aquí termina el párrafo**.**’”
/ . $/	“’?’, ‘!’”
/ \b on \b /	“’Internacional’ → ‘Internacional’”
/ casa | caza /	“Ir a la caza de una casa’”

Ejercicio (1)

Veamos el siguiente ejemplo para entender mejor cómo opera el uso de las expresiones regulares. Veamos la siguiente página:

www.ims.uni-stuttgart.de/projekte/CQPDemos/cqpdemo.html

Se trata de un corpus en inglés construido a partir de algunas novelas escritas por Charles Dickens (1812-

CWB
The IMS Open Corpus Workbench

Este mini-corpus fue elaborado por el *Institute for Natural Language Processing* de la Universidad de Stuttgardt:

www.ims.uni-stuttgart.de

Ejercicio (2)

Busquemos algunos personajes de las novelas de Dickens usando expresiones regulares:

1. En *The Posthumous Papers of the Pickwick Club* existen dos que se apellidan Weller. ¿Quiénes son? ¿Qué expresión regular necesitan para identificarlos?

2. Siguiendo el mismo método, distingan cuántos miembros tiene la familia Wardle, en la misma novela.

3. Basándonos en Oliver Twist, ¿qué tienen que ver Artful Dodger, Jack Dawkins y Fagin con la palabra pickpocket?

4. En David Copperfield, ¿cómo puedo saber que la palabra *strong* puede ser un sustantivo, un adjetivo o un apellido?

Ejercicio (3)

Algunos tips: consideren que algunas búsquedas pueden dar mejores resultados si ocupan etiquetas de partes de la oración. El corpus Workbench emplea etiquetas basadas en el proyecto PennTree Bank. Las pueden consultar en la siguiente liga:

www.ims.uni-stuttgart.de/projekte/CQPDemos/CQPDemo/help-tagset.html

Pensando en estas etiquetas, ustedes pueden formular patrones como los siguientes:

```
gentleman

un*ness

Oliver Twist

said NP

as A as NP

PP=from PP=to

It|it was [ Det ] N of { A } N
```

¡OJO!: el signo " | ", en el buscador del Workbench equivale a disyunción (*x OR y*)

Usando gramáticas con expresiones regulares (1)

Siguiendo este proceso propuesto en el Corpus Workbench, también podemos aplicarlo para crear gramáticas que describan patrones sintácticos concretos, y obtener resultados pertinentes.
NLTK nos permite hacer esto. Veamos de primera instancia la siguiente demo:

```
>>> import nltk
>>> import nltk.chunk.regexp.demo()
```

Este es un programa de prueba que realiza análisis de frases usando una gramática libre de contexto, así como métodos estadísticos para evaluar qué análisis es el más adecuado para describir tales frases.

Usando gramáticas con expresiones regulares (2)

Ahora veamos si podemos aplicar esto a un texto real. Primero, requerimos de un texto, el cual vamos a obtener de la WEB, usando el siguiente proceso:

```
>>> import nltk
>>> import re
>>> import os
>>> url01 = "http://news.bbc.co.uk/2/hi/health/2284783.stm"
>>> html01 = urlopen(url01).read()
>>> html01 [:60]
```

Usando gramáticas con expresiones regulares (2)

Una vez que hemos accedido a nuestro texto WEB, ahora requerimos editarlo para pasarlo a un formato que sea legible para los analizadores lingüísticos de NLTK. Entonces, el primer paso es la tokenización:

```
>>> raw_text01 = nltk.clean_html(html01)
>>> tokens01 = nltk.word_tokenize(raw_text01)
>>> tokens01 [:60]
```

Usando gramáticas con expresiones regulares (3)

En el objeto *tokens01* ya podemos hacer algunas tareas básicas, p. e., buscar colocaciones:

```
>>> Text_Gene01 = nltk.Text(tokens01)
>>> Text-Gene01.concordance('gene')
>>> Text-Gene01.concordance('disease')
>>> Text-Gene01.concordance('cancer')
```

Usando gramáticas con expresiones regulares (4)

Empero, también puedo implementar recursos como un analizador de frases, para lo cual requerimos primero que nuestro corpus esté etiquetado. Aplicando el siguiente proceso, cubrimos esta fase:

```
>>> Tagged_Gene01 = nltk.pos_tag(tokens01)
>>> Tagged_Gene01 [:60]
```

Usando gramáticas con expresiones regulares (5)

Para concluir, implementemos una gramática que haga uso de expresiones regulares para hacer búsqueda de patrones regulares en frases nominales. Veamos este ejemplo:

```
>>> Tagged_Gene01 [:13]
>>> sentence01 = # Resultado de Tagged_Gene01 [:13]
>>> Tagged_Gene01 [15:30]
>>> sentence02 = # Resultado de Tagged_Gene01 [15:30]
>>> Tagged_Gene01 [31:50]
>>> sentence03 = # Resultado de Tagged_Gene01 [31:50]
```

Usando gramáticas con expresiones regulares (6)

Ahora, parea hacer la búsqueda, diseñamos nuestra gramática de la manera siguiente:

```
>>> rule01 = "NP: {<DT>?<JJ>*<NN>}"
>>> rule02 = "NP: {<NNP>*}"
>>> rule03 = "NP: {<NNS>*}"
>>> cp = nltk.RegexpParser(rule01, rule2, rule3)
>>> Results01 = cp.parse(sentence01, sentence 2, sentence 3)
>>> print Results01
```

Gracias por su atención

Blog del curso: http://discurso-uaq.weebly.com/