PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

FACULTAD DE LETRAS

Introducción a la lingüística computacional

César Antonio Aguilar

Facultad de Lenguas y Letras

05/10/2017

Cesar.Aguilar72@gmail.com

Síntesis de la clase pasada (1)

Vimos el **algoritmo de Brill**, el cual nos permite a relacionar palabras con sus categorías gramaticales, representadas como etiquetas.

```
>>> nltk.tag.brill.demo()
Training Brill tagger on 80 sentences...
Finding initial useful rules...
    Found 6555 useful rules.

           B      |
   S   F   r   O  |        Score = Fixed - Broken
   c   i   o   t  |  R     Fixed = num tags changed incorrect -> correct
   o   x   k   h  |  u     Broken = num tags changed correct -> incorrect
   r   e   e   e  |  1     Other = num tags changed incorrect -> incorrect
   e   d   n   r  |  e
------------------+-------------------------------------------------------
  12  13   1   4  | NN -> VB if the tag of the preceding word is 'TO'
   8   9   1  23  | NN -> VBD if the tag of the following word is 'DT'
   8   8   0   9  | NN -> VBD if the tag of the preceding word is 'NNS'
   6   9   3  16  | NN -> NNP if the tag of words i-2...i-1 is '-NONE-'
   5   8   3   6  | NN -> NNP if the tag of the following word is 'NNP'
   5   6   1   0  | NN -> NNP if the text of words i-2...i-1 is 'like'
   5   5   0   3  | NN -> VBN if the text of the following word is '*-1'
   ...
>>> print(open("errors.out").read())
             left context |    word/test->gold     | right context
--------------------------+------------------------+--------------------------
                          |      Then/NN->RB       | ,/, in/IN the/DT guests/N
, in/IN the/DT guests/NNS |       '/VBD->POS       | honor/NN ,/, the/DT speed
'/POS honor/NN ,/, the/DT |    speedway/JJ->NN     | hauled/VBD out/RP four/CD
NN ,/, the/DT speedway/NN |     hauled/NN->VBD     | out/RP four/CD drivers/NN
DT speedway/NN hauled/VBD |      out/NNP->RP       | four/CD drivers/NNS ,/, c
dway/NN hauled/VBD out/RP |      four/NNP->CD      | drivers/NNS ,/, crews/NNS
hauled/VBD out/RP four/CD |    drivers/NNP->NNS    | ,/, crews/NNS and/CC even
P four/CD drivers/NNS ,/, |     crews/NN->NNS      | and/CC even/RB the/DT off
NNS and/CC even/RB the/DT |    official/NNP->JJ    | Indianapolis/NNP 500/CD a
                          |     After/VBD->IN      | the/DT race/NN ,/, Fortun
ter/IN the/DT race/NN ,/, |    Fortune/IN->NNP     | 500/CD executives/NNS dro
s/NNS drooled/VBD like/IN |  schoolboys/NNP->NNS   | over/IN the/DT cars/NNS a
olboys/NNS over/IN the/DT |      cars/NN->NNS      | and/CC drivers/NNS ./.
```

Síntesis de la clase pasada (2)

Veamos un ejemplo de cómo trabaja Brill. Para ello, vamos a ir al siguiente sitio:

http://cst.dk/online/pos_tagger/uk/index.html

El cual presenta una versión **demo**, desarrollada por el **Centro para Tecnologías del Lenguaje**, de la Universidad de Copenhagen, en Dinamarca. Veamos qué pasa con nuestro artículo de Mira Ariel (u otro texto que tengan a la mano):

Centre for Language Technology
Department of Nordic Studies and Linguistics

http://cst.ku.dk/english/

Síntesis de la clase pasada (3)

Y el resultado es:

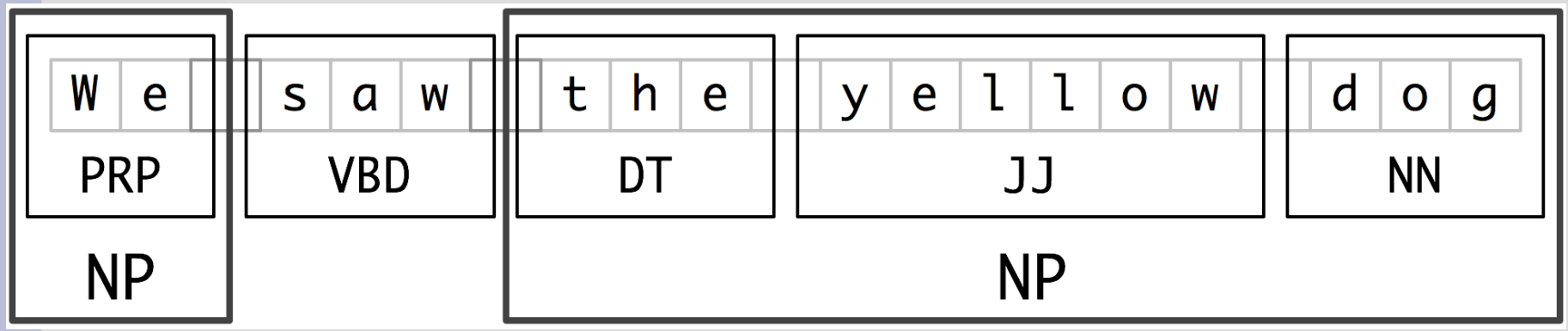
Output

(THE/DT/(THE DEVELOPMENT/VBG/DEVELOPMENT OF/DT/OF PERSON/JJ/PERSON AGREEMENT/NN/AGREEMENT MARKERS/NNS/MARKER :/:/:
FROM/IN/FROM PRONOUNS/NNS/PRONOUN TO/TO/to HIGHER/JJR/HIGHER ACCESSIBILITY/NNP/ACCESSIBILITY MARKERS*/NNP/MARKER
Mira/NNP/Mira Ariel/NNP/Ariel (/(/(1999/CD/1999)/)/)
Tel-Aviv/NNP/Tel-Aviv University/NNP/University
1/LS/1 ././.
From/IN/From free/JJ/free pronouns/NNS/pronoun to/TO/to verbal/JJ/verbal agreement/NN/agreement :/:/:
Introduction/NN/Introduction
Grammatical/JJ/Grammatical items/NNS/item often/RB/often begin/VBP/begin their/PRP$/their linguistic/JJ/linguistic life/NN/life as/IN/as regular/JJ/regular lexical/JJ/lexical items/NNS/item (/(/(Meillet/NNP/Meillet 1912/CD/1912)/)/)
././.
Grammaticization/NN/Grammaticization is/VBZ/be said/VBD/say to/TO/to have/VB/have occurred/VBN/occur when/WRB/when the/DT/the position/NN/position of/IN/of such/JJ/such lexemes/NNS/lexemes (/(/(or/CC/or even/JJ/even phrases/NNS/phrase)/)/) becomes/VBZ/become fixed/VBN/fix ,/,/, their/PRP$/their meaning/NN/meaning is/VBZ/be generalized/bleached/VBN/generalized/bleached ,/,/, their/PRP$/their domain/NN/domain of/IN/of applicability/NN/applicability enlarged/JJ/enlarge ,/,/, significantly/RB/significantly raising/VBG/raise their/PRP$/their frequency/NN/frequency ,/,/, their/PRP$/their form/NN/form modified/VBN/modify (/(/(usually/RB/usually phonetically/RB/phonetically reduced/VBN/reduce)/)/) ,/,/, and/CC/and their/PRP$/their occurrence/NN/occurrence made/VBN/make obligatory/JJ/obligatory ,/,/, even/RB/even when/WRB/when informationally/RB/informationally redundant/JJ/redundant (/(/(see/VB/see Bybee/NNP/Bybee et/FW/et al/NNS/al ././.
1994/CD/1994)/)/) ././.

Análisis gramatical con NLTK (1)

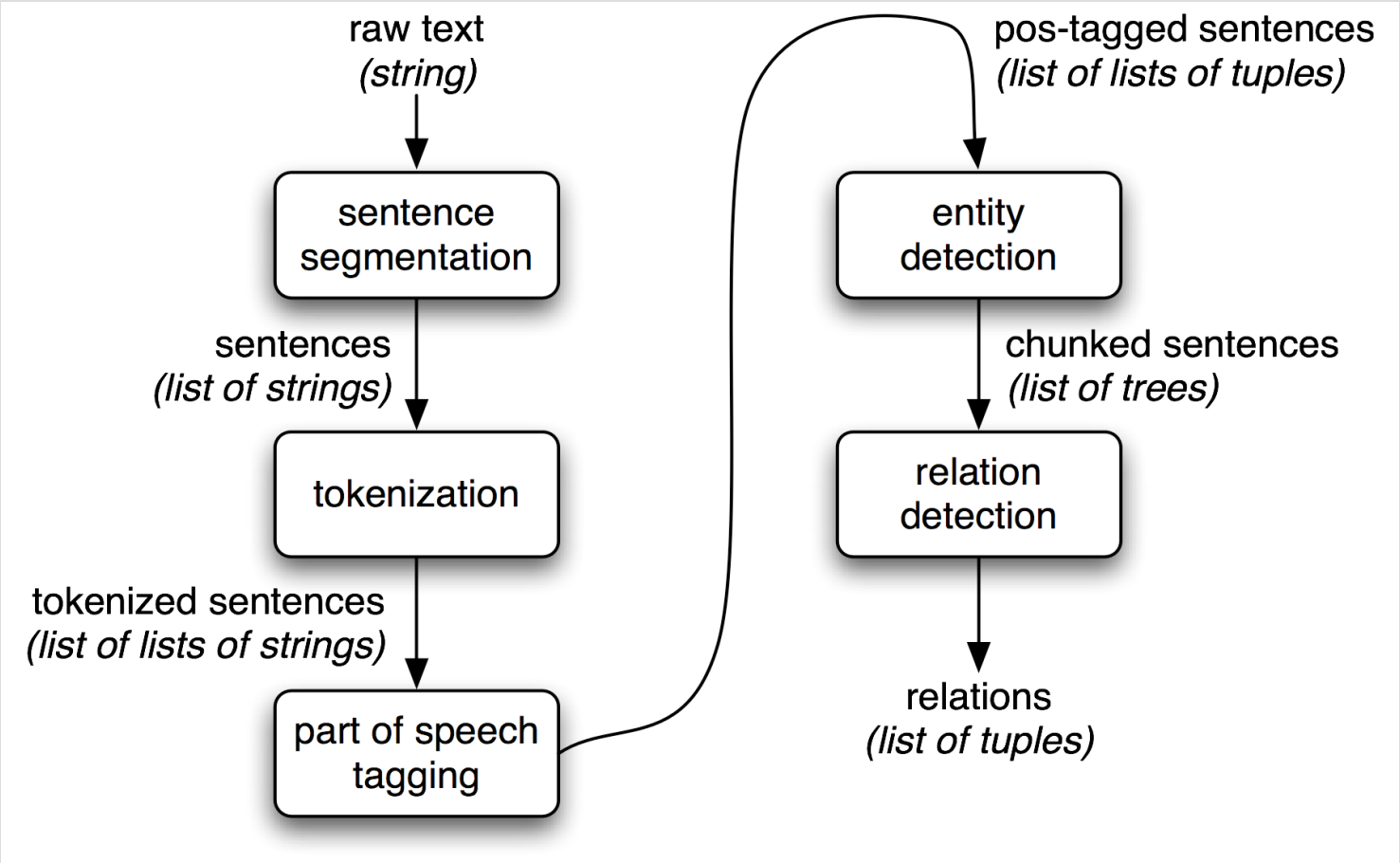
Retomando el trabajo que hicimos etiquetando corpus en inglés, la siguiente tarea que podemos realizar es tratar de identificar frases y oraciones dentro de tales documentos.

A este proceso usualmente se le conoce como *chunking* y *parsing*. El término *chunking* se refiere al proceso de identificación de frases, mientras que el de *parsing* alude al proceso de detección, análisis y graficado de oraciones. De momento, nos concentramos en los métodos de *chunking* que ofrece NLTK.



Análisis gramatical con NLTK (2)

En términos más simples, lo que estamos haciendo es seguir una **cadena de procesamiento**, la cual podemos describir del siguiente modo:



Breve introducción a teorías sintácticas (1)

De acuerdo con Jurafsky y Martin (2007), el estudio de la sintaxis es una de esas áreas en la lingüística con un abolengo (o *pedigree) muy* elevado. Hay varias razones para esto:

1. La sintaxis tiene una larga historia: por lo menos de unos 2000 años, si consideramos la gramática que Panini delineó para el sánscrito.

2. Con respecto otras ramas de la lingüística, es quizá la que más se ha trabajado desde un sinnúmero de enfoques: normativo, descriptivo, predictivo, formal, funcional, cognitivo, estadístico, y un largo etc.

3. A partir de los años 60, Chomsky y otros la han ponderado como el nivel de composición más importante de una lengua natural.

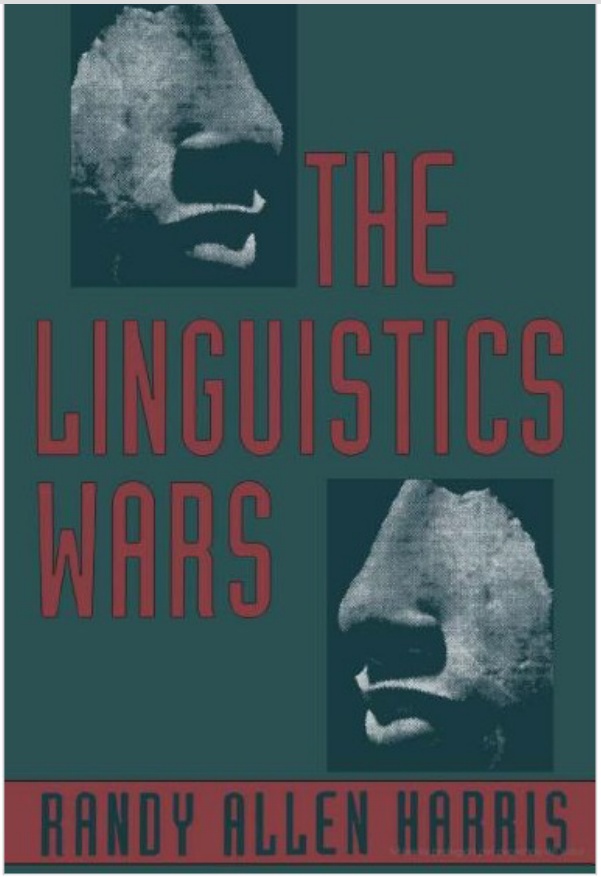
4. Considerando el punto 3, hasta el momento parece ser que una de las mejores evidencias que tenemos para reconocer una lengua natural (o humana) de otros sistemas comunicativos.

Breve introducción a teorías sintácticas (2)

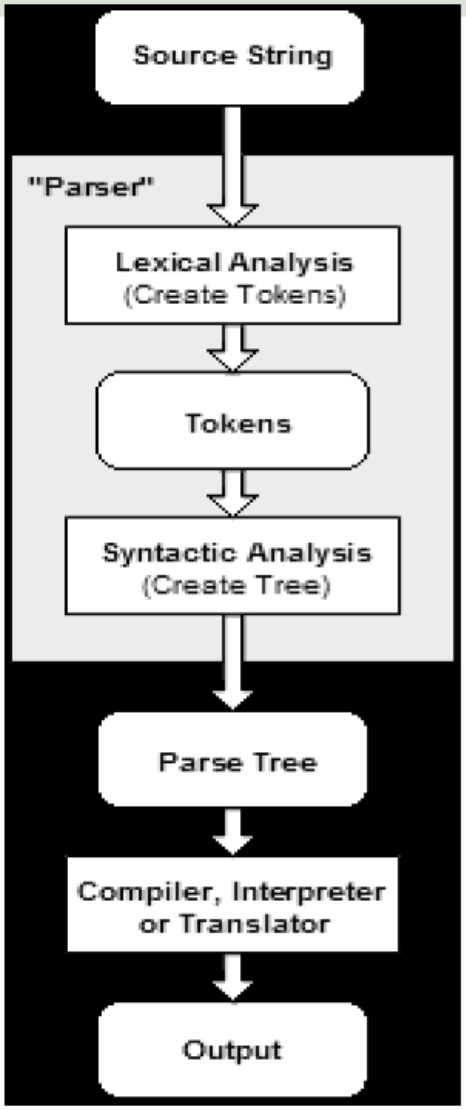
Ahora bien, más allá de todos estos honores y medallas para los estudios sintácticos, por que le interesa al PLN meterse en estas cosas?

1. **Por mera cuestión de ego:** digamos que si de por sí esta área es una de las más conflictivas de la lingüística, los de PLN quisieran demostrar que también pueden librar "las grandes batallas" dentro de la lingüística.

2. **Porque la sintaxis es realmente un nivel importante:** hasta el momento, lo que hemos visto en este curso es constituir corpus, y hacer análisis al nivel de palabras. Empero, nos falta dar el salto a problemas sobre cómo analizar frases y oraciones.



Breve introducción a teorías sintácticas (3)



3. **Por cuestiones teóricas:** tomando en cuenta la influencia que ha tenido las gramáticas formales en la inteligencia artificial, en PLN se han desarrollado teorías sintácticas que permiten modelar, predecir y generar frases y oraciones.

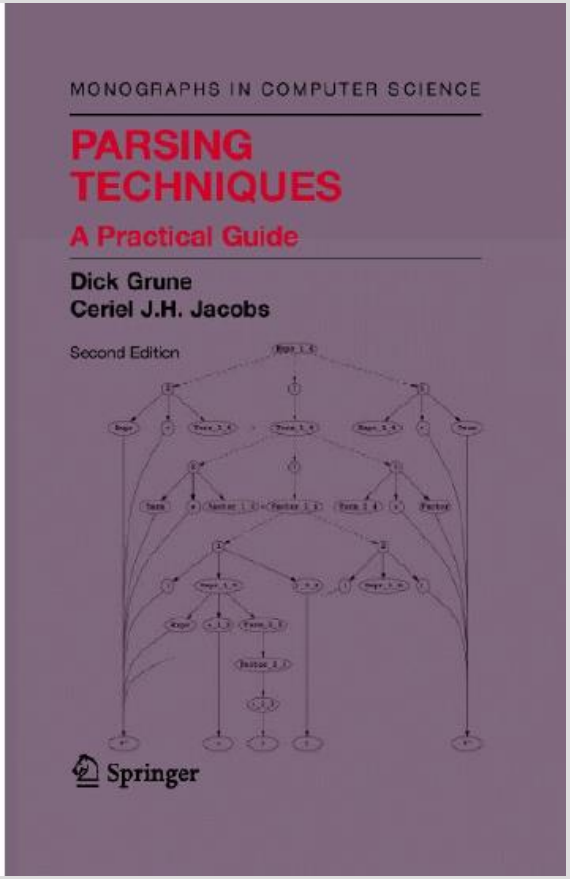
4. **Por cuestiones prácticas:** varias ramas de la ingeniería lingüística requieren de módulos de procesamiento sintáctico.

Breve introducción a teorías sintácticas (4)

Si atendemos los puntos 3 y 4 de la lamina anterior, entonces tenemos dos puntos de vista que valen la pena considerar:

En un plano teórico:

1. El parsing es una tarea de PLN que se relaciona con la validación de hipótesis sobre cómo operan las lenguas.
2. El parsing implica una serie de procesos heurísticos que sustentan el poder de creativo del lenguaje.
3. La sintaxis, hasta hoy, es quizá la parte más susceptible a un análisis matemático, junto con la fonética y la fonología.



Breve introducción a teorías sintácticas (5)

En un plano práctico:

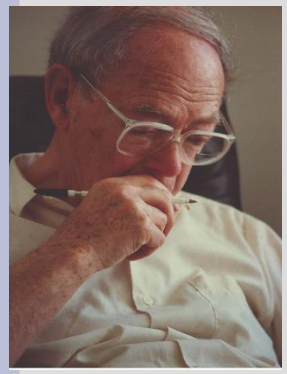
1. El parsing supone el uso de técnicas y métodos tales como con el reconocimiento de patrones, aprendizaje-máquina, modelos heurísticos, modelos estocásticos, etc.

2. Se ocupa en tareas tales como extracción de información, minería de textos, extracción terminológica, reconocimiento de entidades nombradas, ontologías, etc.

INCIDENT TYPE	bombing
DATE	March 19
LOCATION	El Salvador: San Salvador (city)
PERPETRATOR	urban guerrilla commandos
PHYSICAL TARGET	power tower
HUMAN TARGET	-
EFFECT ON PHYSICAL TARGET	destroyed
EFFECT ON HUMAN TARGET	no injury or death
INSTRUMENT	bomb

Breve introducción a teorías sintácticas (6)

En PLN hay muchos autores que se interesan —tanto por cuestiones teóricas y/o prácticas— en el desarrollo de gramáticas formales.



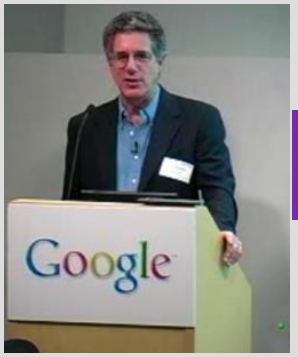
Zellig Harris (1909-1992)

Noam Chomsky (1928)

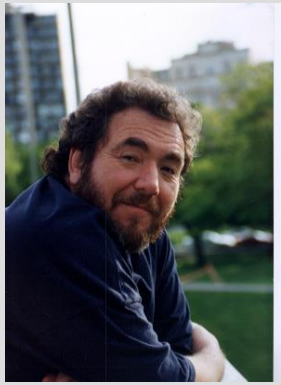
Igor Mel'čuk (1932)

Aravind K. Joshi (1929)

Joan Bresnan (1945)

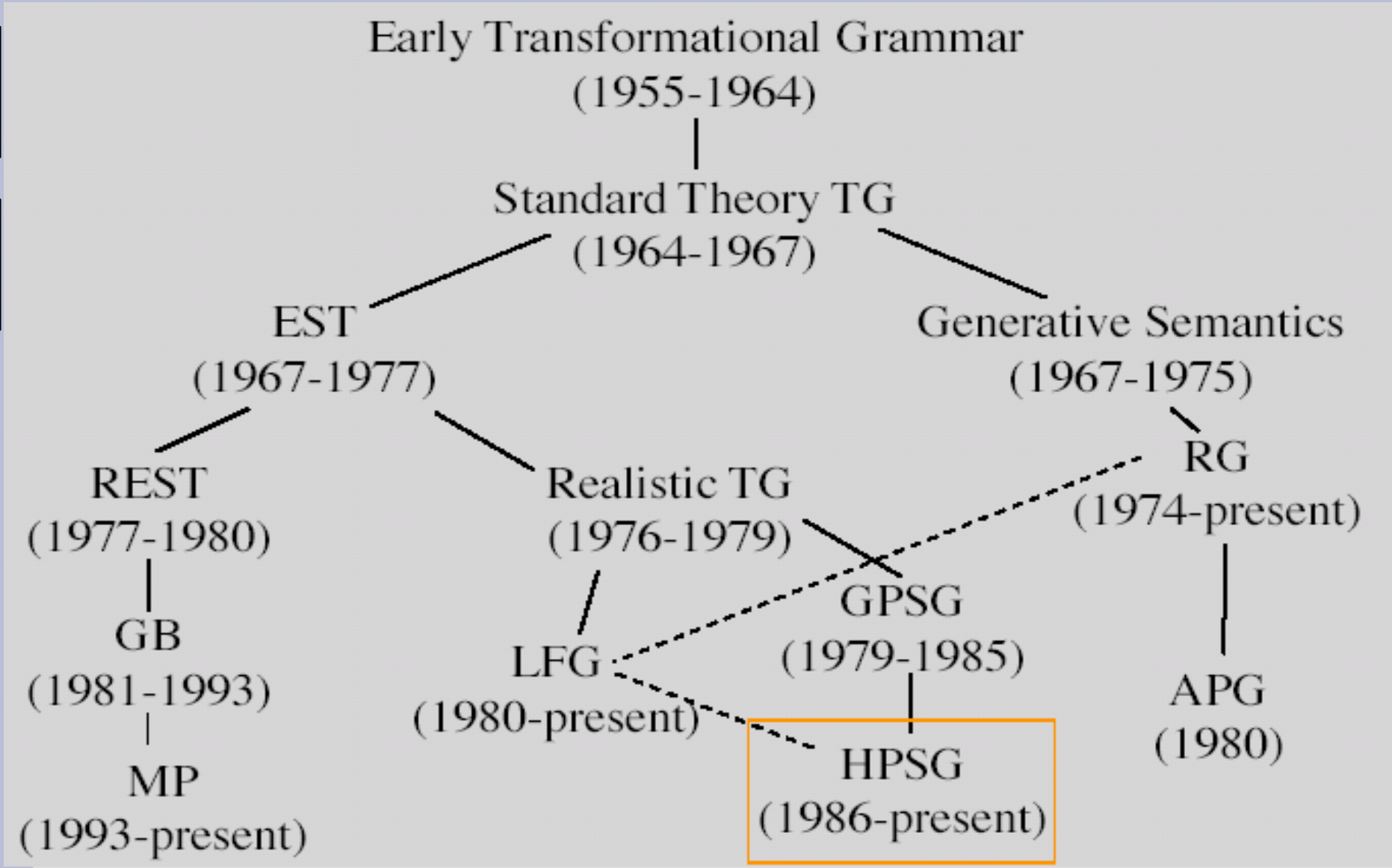


Ray Jackendoff (1945)



Ivan Sag (1949)

La familia “G”


Early Transformational Grammar
(1955-1964)
Standard Theory TG
(1964-1967)
EST
(1967-1977)
Generative Semantics
(1967-1975)
REST
(1977-1980)
Realistic TG
(1976-1979)
RG
(1974-present)
GB
(1981-1993)
LFG
(1980-present)
GPSG
(1979-1985)
APG
(1980)
MP
(1993-present)
HPSG
(1986-present)

Interludio…

Pero antes de seguir, analicemos estas preguntas:

¿Qué es una oración?

¿Qué es una frase?

¿Qué es una gramática?

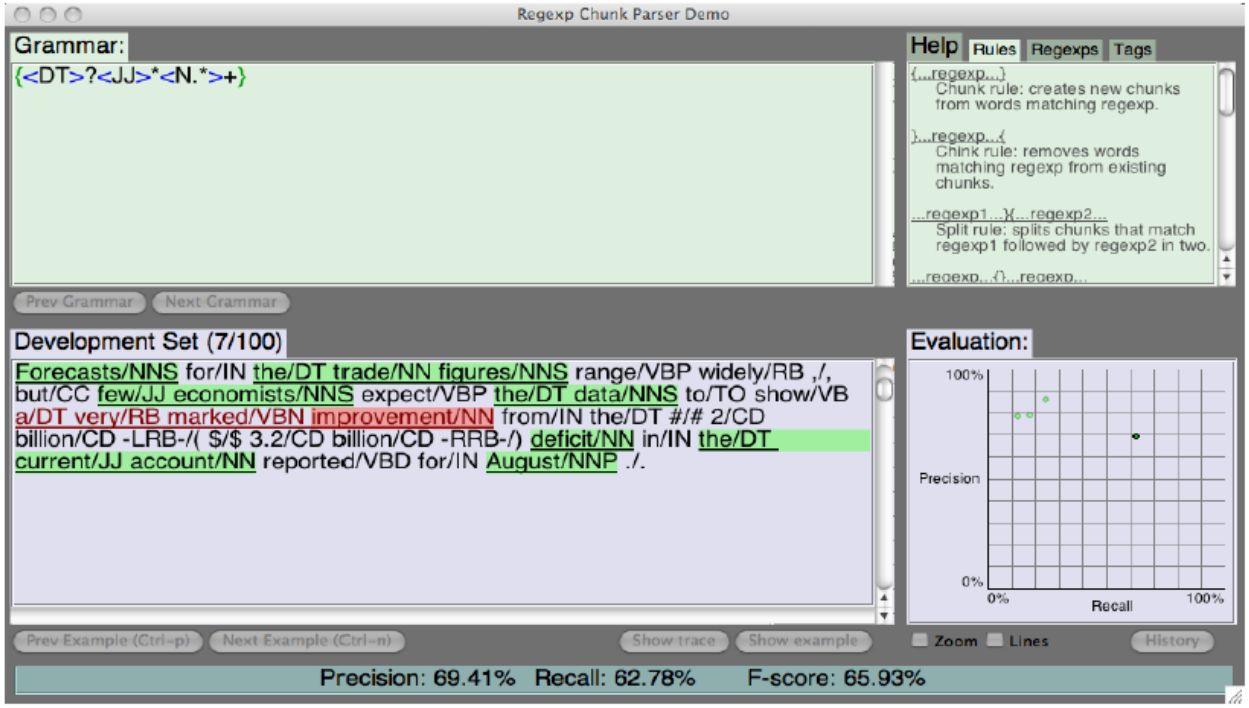
¿Hay relaciones básicas en la sintaxis de una lengua natural? ¿Cuáles son?

¿La respuesta que hagamos a estas preguntas, ¿está influida por la teoría sintáctica que tengamos en mente?

Frase = estructuras formadas por palabras (1)

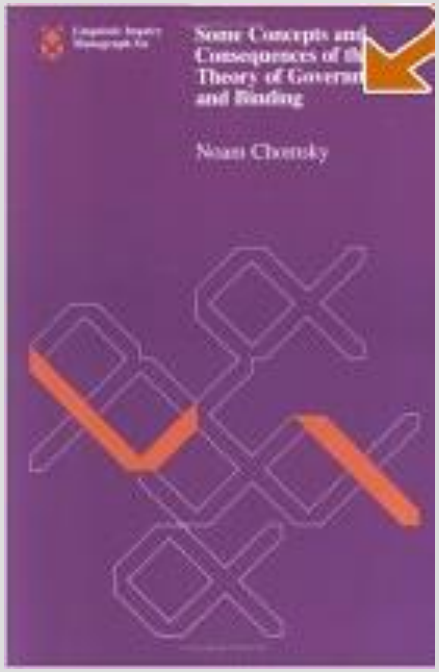
Basándonos en nuestro conocimiento sobre etiquetado POS, podemos suponer que una secuencia de palabras, asociada conforme a un patrón regular, confugira una potencial estructura de frase o de oración.

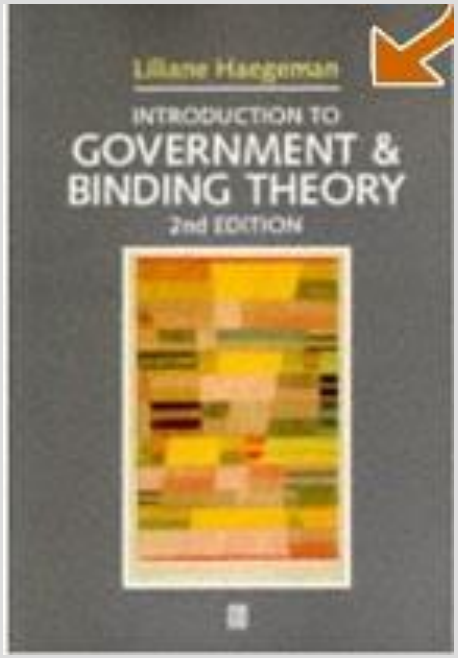
Lo que intentamos hacer entonces es delinear, analizar y evaluar una regla formal que dé cuenta de tal patrón de constitución.

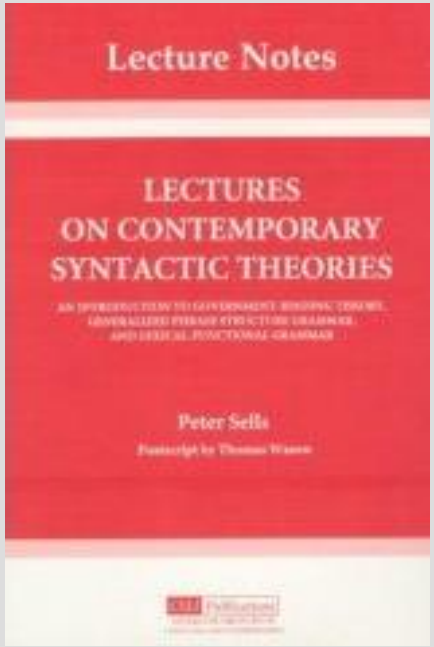


Frase = estructuras formadas por palabras (2)

En aras de responder a las preguntas que hemos planteado, vamos a analizar algunos conceptos básicos formulados por un modelo de gramática formal llamado **teoría de la rección y el ligamiento** (inglés: *Government and Binding Theory*, o GB), desarrollada por el lingüista norteamericano Noam Chomsky en los años 80, la cual ha tenido gran impacto hasta el día de hoy.







Frase = estructuras formadas por palabras (3)

- GB es un modelo de descripción gramatical que, a partir de una serie de componentes y principios generales, permite explicar el comportamiento de cualquier estructura sintáctica, situada dentro de un parámetro determinado.

- Considerando el rasgo anterior, GB ofrece pistas valiosas en torno a la existencia de una gramática universal, la cual subyace en todas las lenguas, sin que dicha gramática se vea afectada crucialmente por el parámetro en el cual se desenvuelve una lengua particular.

- De este modo, más allá de ofrecer un conjunto de reglas y transformaciones particulares para una lengua, lo que se busca es dar una descripción precisa de fenómenos que son comunes a varias (o todas) las lenguas humanas, empleando un mínimo de componentes y principios.

Frase = estructuras formadas por palabras (4)

Clases de palabras (1)

Una palabra es el elemento mínimo constituyente de una estructura de frase o de oración. Equivale a *un carácter* dentro de una cadena de n-dimensiones (esto es, podemos agrupar caracteres hasta tener una frase, un enunciado o un conjunto de enunciados).

Una palabra puede tener una estructura formal y una información semántica particular. Por ejemplo, *Neoliberal* es una construcción compuesta por una raiz *liber* y dos afijos, *Neo-* (prefijo) y *-al* (sufijo), y contiene información léxica que le permite desempeñarse como adjetivo (*El presidente de México es neoliberal*), o como un nombre (*Un neoliberal gobierna a México desde el 2012*), cuyo contenido referencial hace alusión a una tendencia o afiliación política, etc.

Clases de palabras (2)

1. Nombres (Juan, mesa, hombre-rana, autorregulación, etc.).

2. Verbos (Correr, hacer, comprar, regalar, etc.).

3. Determinates (El, una, algunos, aquel, todos, mi, nuestro, etc.).

4. Adjetivos (rojo, alto, legal, cuatro, etc.).

5. Adverbios (muy, bastante, ayer, allá, etc.).

6. Preposiciones (a, ante, bajo, cabe, con, contra, etc.).

7. Conjunciones (y, o, pero, sin embargo, si... entonces, etc.).

8. Pronombres y clíticos (yo, tú, mío, suyo, le, les, se).

Clases de palabras (3)

Table 2:
The Penn Treebank POS tagset

1.	CC	Coordinating conjunction	25.	TO	*to*
2.	CD	Cardinal number	26.	UH	Interjection
3.	DT	Determiner	27.	VB	Verb, base form
4.	EX	Existential *there*	28.	VBD	Verb, past tense
5.	FW	Foreign word	29.	VBG	Verb, gerund/present participle
6.	IN	Preposition/subord. conjunction	30.	VBN	Verb, past participle
7.	JJ	Adjective	31.	VBP	Verb, non-3rd ps. sing. present
8.	JJR	Adjective, comparative	32.	VBZ	Verb, 3rd ps. sing. present
9.	JJS	Adjective, superlative	33.	WDT	*wh*-determiner
10.	LS	List item marker	34.	WP	*wh*-pronoun
11.	MD	Modal	35.	WP$	Possessive *wh*-pronoun
12.	NN	Noun, singular or mass	36.	WRB	*wh*-adverb
13.	NNS	Noun, plural	37.	#	Pound sign
14.	NNP	Proper noun, singular	38.	$	Dollar sign
15.	NNPS	Proper noun, plural	39.	.	Sentence-final punctuation
16.	PDT	Predeterminer	40.	,	Comma
17.	POS	Possessive ending	41.	:	Colon, semi-colon
18.	PRP	Personal pronoun	42.	(	Left bracket character
19.	PP$	Possessive pronoun	43.	)	Right bracket character
20.	RB	Adverb	44.	"	Straight double quote
21.	RBR	Adverb, comparative	45.	‘	Left open single quote
22.	RBS	Adverb, superlative	46.	“	Left open double quote
23.	RP	Particle	47.	’	Right close single quote
24.	SYM	Symbol (mathematical or scientific)	48.	”	Right close double quote

Frase = estructuras formadas por palabras (5)

Toda palabra dentro de la estructura de una frase cumple únicamente dos funciones: o ser el **núcleo de ésta, o ser un modificador.**

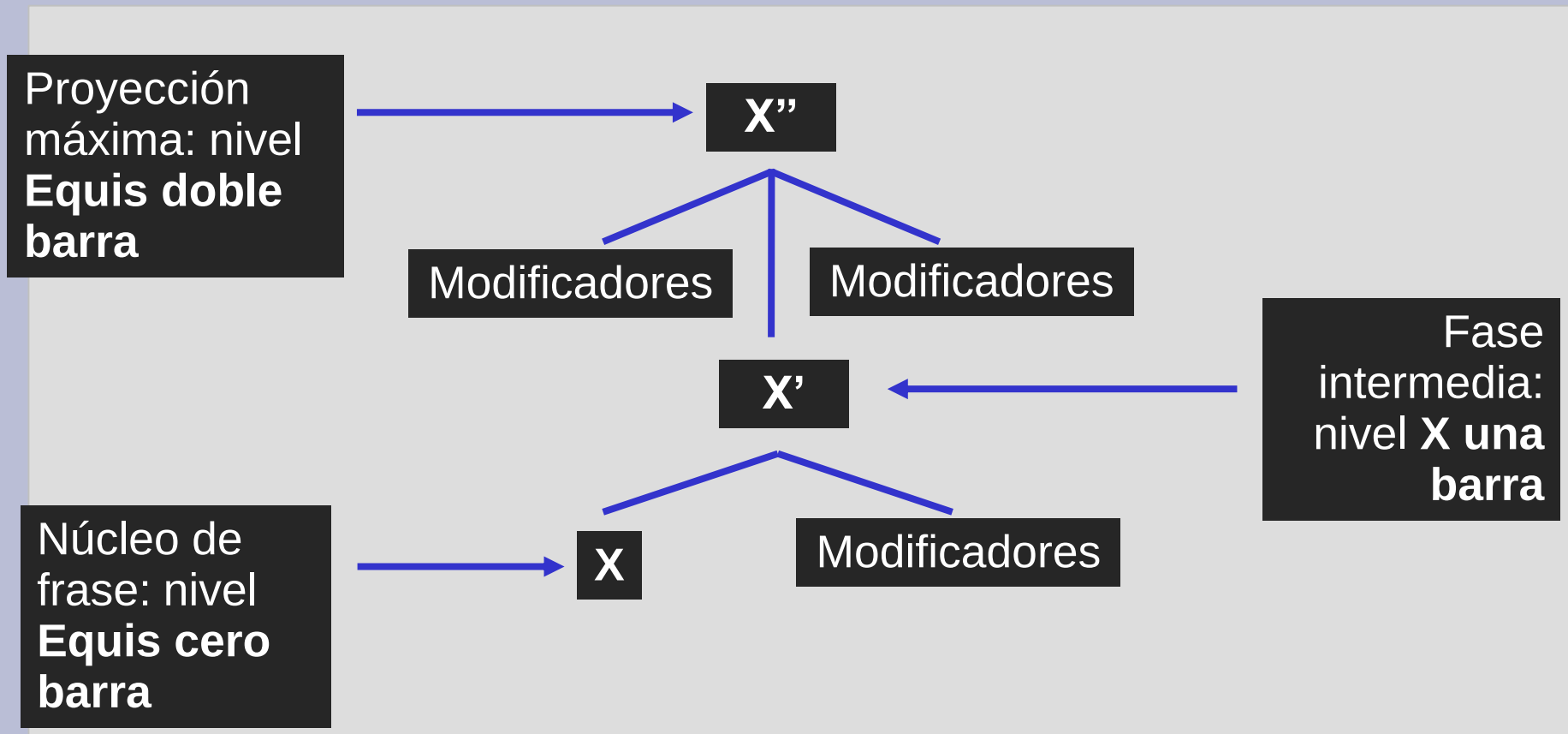
Así, la parte central (esto es, la *cabeza*) de una frase es entonces su núcleo, de modo que éste tiene la propiedad de lograr una **proyección máxima**, dependiendo de cuántos modificadores se le puedan agregar.

Por lo general, este nivel de proyección máxima se logra cubriendo tres fases:

1. Una fase 0, en la que simplemente contamos con un núcleo aislado. At esta fase se le conoce como **nivel X^0** (equis cero barra).
2. Una fase intermedia que configura una estructura conformada por el núcleo más un primer modificador, que puede ser una palabra, una frase, una oración o varias oraciones. A este nivel se le conoce como **nivel X'** (equis una barra).
3. Una fase final en donde la frase se completa en su totalidad, al añadirse un modificador más. Este nivel se le conoce como **nivel X''** (equis doble barra).

Frase = estructuras formadas por palabras (6)

El esquema es el siguiente:



Etiquetado sintáctico y n-gramas (1)

El análisis de frases en NLTK, como hemos visto, se apoya en varios recursos, en aras de ser eficaz y confiable a la hora de explorar y obtener datos de un corpus. Un recurso que podemos emplear para mejorar los niveles de Precision & Recall es el uso de **n-gramas**.

<s> I
I want
want to
to eat
eat Chinese
Chinese food
food </s>

Los *n-gramas* son secuencias de palabras que muestran una alta recurrencia en su relación, de modo que pueden convertirse en **tuplas**, **tripletas, tetrapletas, etc.** En otras palabras, nos referimos a **unigramas, bigramas, trigramas, tetragramas**..., y demás secuencias similares.

Chunking y n-gramas (2)

Para hacer un analizador de frases con NLTK, nos vamos a basar en el siguiente código, el cual pueden localizar en la siguiente página de NLTK (en concreto, el ejemplo 7.8):

http://nltk.googlecode.com/svn/trunk/doc/book/ch07.html

El método es el siguiente:

```
class UnigramChunker(nltk.ChunkParserI):
    def __init__(self, train_sents): [1]
        train_data = [[(t,c) for w,t,c in nltk.chunk.tree2conlltags(sent)]
                      for sent in train_sents]
        self.tagger = nltk.UnigramTagger(train_data) [2]

    def parse(self, sentence): [3]
        pos_tags = [pos for (word,pos) in sentence]
        tagged_pos_tags = self.tagger.tag(pos_tags)
        chunktags = [chunktag for (pos, chunktag) in tagged_pos_tags]
        conlltags = [(word, pos, chunktag) for ((word,pos),chunktag)
                     in zip(sentence, chunktags)]
        return nltk.chunk.conlltags2tree(conlltags)
```

Chunking y n-gramas (3)

Este código es un buen ejemplo de lo que es una programación orientada a objetos. Así, definimos una clase, la de **unigramas**, la cual define dos métodos: el primer método es un constructor de n-gramas:

```
class UnigramChunker(nltk.ChunkParserI):
    def __init__(self, train_sents):
        train_data = [[(t,c) for w,t,c in nltk.chunk.tree2conlltags(sent)]
                      for sent in train_sents]
        self.tagger = nltk.UnigramTagger(train_data)
```

Lo que estamos haciendo aquí es establecer que hay una clase llamada **UnigramChunker**, la cual agrupa a todos aquellos unigramas que estén constituidos por palabras y etiquetas, los cuales aparezcan en oraciones pertenecientes al corpus CoNLL 2000.

Chunking y n-gramas (4)

La segunda parte del método es precisamente nuestro *chunker*, el cual agrupa todas las palabras que tengan una etiqueta morfosintáctica, que puedan formar parte de un unigrama:

```
 def parse(self, sentence): [3]
        pos_tags = [pos for (word,pos) in sentence]
        tagged_pos_tags = self.tagger.tag(pos_tags)
        chunktags = [chunktag for (pos, chunktag) in
tagged_pos_tags]
        conlltags = [(word, pos, chunktag) for
((word,pos),chunktag)
                    in zip(sentence, chunktags)]
    return nltk.chunk.util.conlltags2tree(conlltags)
```

Chunking y n-gramas (5)

Retomemos nuestra gramática para identificar frases nominales, con la diferencia de que ahora consideramos patrones codificados en forma de unigramas. Nuestro código es:

```
>>> test_sents = conll2000.chunked_sents('test.txt',
chunk_types=['NP'])
>>> train_sents = conll2000.chunked_sents('train.txt',
chunk_types=['NP'])
>>> unigram_chunker = UnigramChunker(train_sents)
>>> print unigram_chunker.evaluate(test_sents)
```

Chunking y n-gramas (6)

Y el resultado es:

```
ChunkParse score:
    IOB Accuracy:   92.9%
    Precision:      79.9%
    Recall:         86.8%
    F-Measure:      83.2%
```

Traduciendo esto, lo que quiere decir es que si bien bajó nuestro nivel de recuperación (86.8%), logramos subir en un 9.9% nuestro grado de precisión, lo que permite que tengamos un balance del 83.2 %: parece que el uso de unigramas nos ayuda mejorar nuestra búsqueda de FNs.

Chunking y n-gramas (7)

¿Y qué pasa si trabajamos mejor con bigramas? Para hacerlo, podemos volver a utilizar nuestro código, cambiando únicamente los siguientes nombres:

1. UnigramChunker >> BigramChunker

2. UnigramTagger >> BigramTagger

Una vez hechos estos cambios, volvemos a aplicar nuestro programa al corpus CoNLL 2000, y obtenemos los siguientes resultados:

Chunking y n-gramas (8)

```
ChunkParse score:
    IOB Accuracy:  93.3%
    Precision:     82.3%
    Recall:        86.8%
    F-Measure:     84.5%
```

Al final, lo que observamos es que nuestro nivel de recuperación se mantiene igual (86.8%), pero hemos logrado aumentar nuestro grado de precisión hasta un 82.3%, lo cual permite lograr un balance más armónico entre los candidatos a FN que encontramos, *versus* aquellos que cuadran de forma muy precisa con nuestra regla.

Gracias por su atención

Blog del curso:
http://cesaraguilar.weebly.com/introduccioacuten-a-la-linguumliacutestica-computacional.html