Un analizador con 'chart' para gramáticas categoriales de unificación

Ignacio Moreno-Torres
Universidad de Málaga.

M. Teresa Solias
Universidad de Barcelona.

I. FUNDAMENTOS DE LAS GRAMATICAS CATEGORIALES

INTRODUCCION HISTORICA

La idea de las gramáticas categoriales se remonta a los estudios realizados por lógicos polacos, cuyos más ilustres representantes en este campo fueron Lésniewski (1929) y, sobre todo, Ajdukiewick (1935).

Estas primeras gramáticas categoriales deben mucho a la Teoría de Tipos de Russell y a la Teoría de las Categorías de Significado de Husserl [1] .

El primero que adaptó esta teoría al lenguaje natural fué Bar-Hiller (1953). Las ventajas que este autor resaltó de los trabajos de sus predecesores fue la precisión y la economía motivadas por el carácter quasi-aritmético de la Aplicación Funcional, su regla principal.

De hecho, en estos primeros trabajos sólo existía la Aplicación Funcional hacia adelante, que veremos en detalle en el siguiente apartado.

En 1958 Lambek recoge los trabajos de Ajdukiewick y Bar-Hillel con la pretensión de construir un sistema formal que permita derivar una serie de reglas o operaciones que, aplicándolas sobre las categorías básicas, permitan

distinguir las oraciones bien formadas de las que no lo estan. De esta manera, las reglas sel sistema que vamos a presentar se derivan formalmente del denominado Cálculo Lambek, que es el sistema de secuentes que Lambek desarrolló. Actualmente Van Benthem (1986,1987,1988) y Moorgart (1988) trabajan en profundidad este tema.

Otro Sistema Formal que ha sido aplicado con éxito a las Gramáticas Categoriales es el de la Lógica Combinatoria de Curry (1930,1960), Curry y Feys (1958) y también Smullyan (1986) . Efectivamente, Steedman (1985,1986,1987,1989) ha mostrado que el uso de los combinadores de esta lógica en Gramáticas Categoriales da excelentes resultados para el análisis del lenguaje natural. En concreto, como veremos, en la resolución de gaps, extracciones y otros fenómenos lingüísticos que son problemáticos para todas las Teorías Lingüísticas.

El inicio del renacimiento actual de la Gramática Categorial se debe a los trabajos de Montague (1973) y de sus seguidores (véase el trabajo de Dowty (1987) sobre Elevación de tipos). Así como los trabajos de Bach (1979), Ades y Steedman (1979, 1982).

GRAMATICAS CATEGORIALES

Una Gramática Categorial (GC) está formada por un conjunto de categorías , un conjunto de operadores que [illegible] para generar las categorías complejas (al menos un

operador) y un conjunto de operaciones (al menos una).

(1) GRAMATICA CATEGORIAL
i. Categorías.
ii. Operadores.
iii. Operaciones o Reglas.

Categorías

En toda GC hay un conjunto de categorías básicas CAT donde :

(2) DEFINICION DE CATEGORIA

(i) Si A es una categoría básica, entonces A pertenece a CAT.

(ii) Si A y B pertenecen a CAT, entonces A/B pertenece a CAT.

NOTA: Aquí '/' vale por cualquiera de los operadores que luego especificaremos.

Operadores

El conjunto de operadores que se usan para formar categorías compuestas se ha ido haciendo más grande a medida que las Gramáticas Categoriales se han ido aplicando a fenómenos del lenguage natural.

(3) OPERADORES

1. *Aplicación funcional por la derecha*

   / X/Y Y --> X

   Ej:

   Det = NP/N

2. *Aplicación funcional por la izquierda*

   \ Y X\Y --> X

Ej: Verbo Intransitivo = S\NP

3. *Aplicación funcional bidireccional*

| X|Y Y --> X, Y X|Y -->X

Ej: Adjetivo = N|N

4. *Producto*

* X Y --> X*Y

)peraciones o reglas

Dependen del tipo de cálculo que se utilice. Las que se :itan a continuación pertenecen a la Gramática Categorial :ombinatoria (Steedman (1987,1989), aunque algunas de ellas ra habían sido introducidas con anterioridad en el lenominado Cálculo Lambek (Lambek (1958)). La Aplicación Funcional es la operación que ya usaban los fundadores de las GC (Ajdukiewicz (1935), Bar-Hillel (1964)) y es usado por todos los seguidores de dichas gramáticas.

:4) OPERACIONES O REGLAS

Aplicación Funcional hacia adelante

X/Y Y -->> X (>) (corresponde al OPERADOR 1)

Aplicación Funcional hacia atrás

Y X\Y --><< X (<) (corresponde al OPERADOR 2)

Ej:

```
El     pitufo    come      estas     moras
NP/N    N      (S\NP)/NP    NP/N       N

----------->              ---------->
    NP                        NP
            ----------------------->
                    S\NP
-------------------------------------->
                  S
```

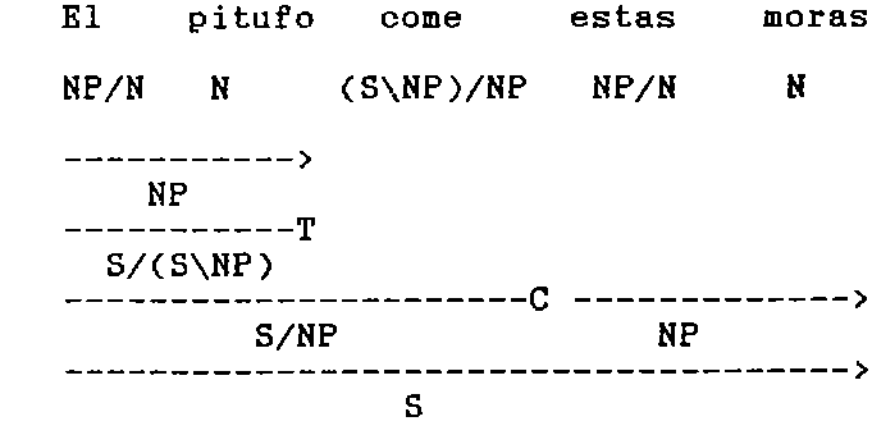
Type-Raising (Regla unária)

X -->T T/(T\X)

Composición hacia adelante

X/Y Y/Z -->B X/Z

Ej:



Composición hacia atrás

Y\Z X\Y -->B X\Z

Sustitución hacia adelante

(X/Y)/Z Y/Z -->S X/Z

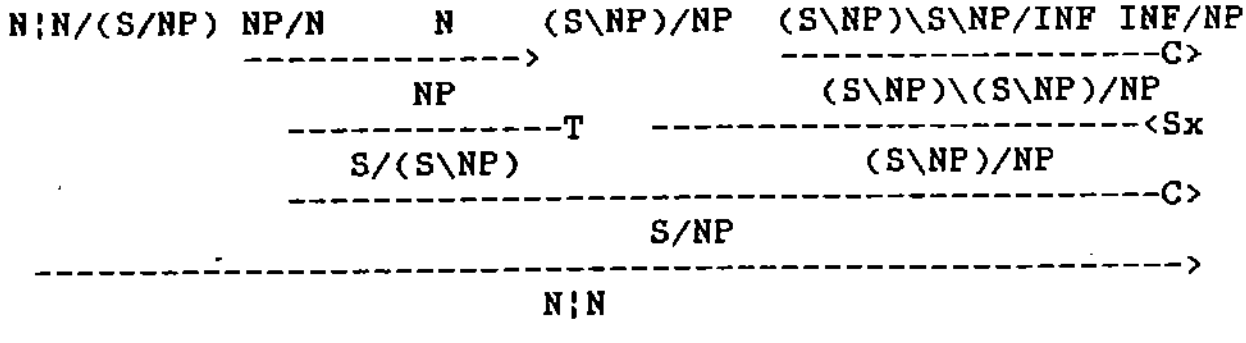
Sustitución hacia atrás

Y\Z (X\Z)\Z -->S X\Z

Sustitución cruzada hacia atrás

Y/Z (X\Y)/Z --> X/Z

Ej: PARASITIC GAPS



CATEGORIAS PARA EL CASTELLANO, EL CATALAN Y OTRAS LENGUAS ROMANICAS

La mayoría de los que han trabajado en Gramáticas Categoriales de Unificación (UCG) o en Gramáticas de Unificación Categoriales (CUG) han desarrollado sus trabajos para el inglés. Por lo tanto las definiciones concretas que ellos han adoptado funcionan para este idioma. Se ha de tener en cuenta que el inglés es una lengua con orden muy fijo y que esa es una propiedad muy valorada desde el punto de vista de las gramáticas categoriales (tambien por una gramática libre de contexto, nótese que para toda gramática categorial existe una Gramática libre de Contexto equivalente, cf. Bach , Pollard (1987)). Sin embargo el castellano y el resto de lenguas románicas poseen un orden semi libre o, como mínimo , contemplan muchas más posibilidades de ordenación que el inglés y demás lenguas de sujeto explícito y sin marcas aparentes de caso.

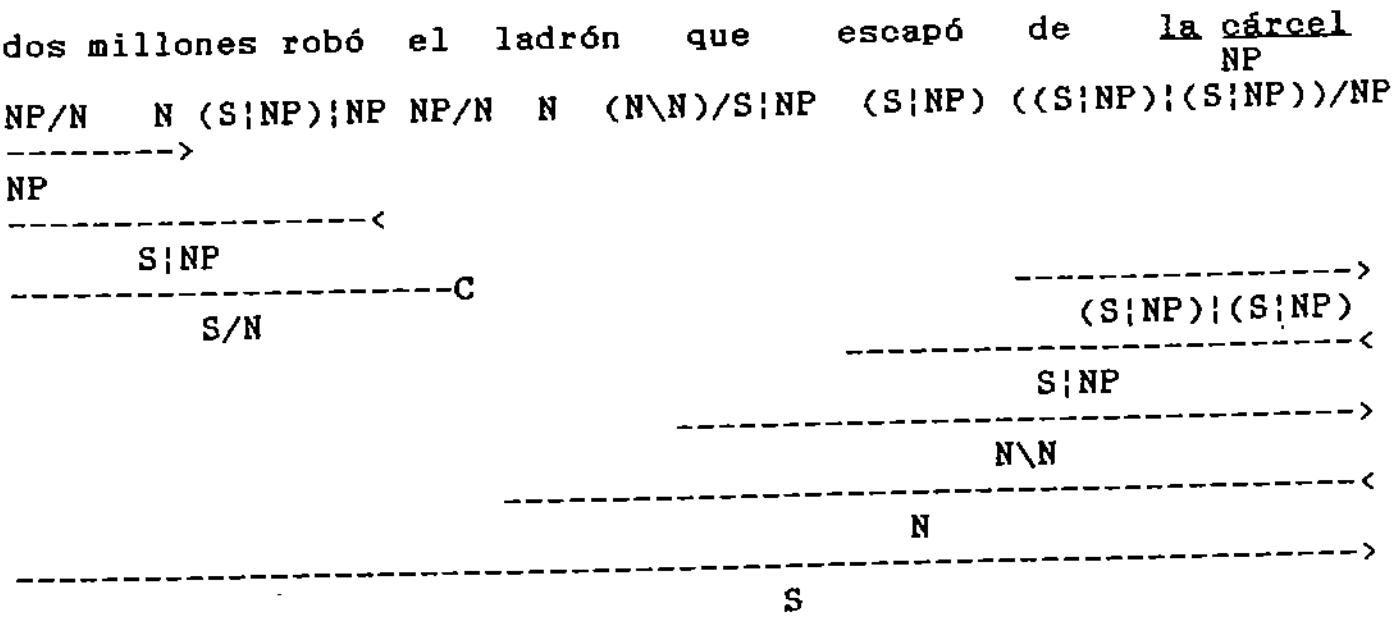
Efectivamente, las lenguas románicas conservan algunas, pocas, marcas de caso, pero tienen una flexión rica que les permite utilizar el sujeto en lugares bien diversos, además de poderlo dejar implícito. Además, los modificadores tambien tienen una libertad de aparición remarcable que necesita ser tratada.

Por estos motivos las categorías que se utilizaban en inglés han tenido que ser modificadas, fundamentalmente en lo que concierne a la dirección de sus barras inclinadas

(slash). Veamos algunos casos de reformulación de esas categorías, que permitirá dar cuenta del orden semi libre del catellano o el catalán, de las extracciones, la posición de algunos adverbios y la inversión libre del sujeto.

	Inglés	**Castellano**
VT	(S\NP)/NP	(S¦NP)¦NP
VI	S\NP	S¦NP
ADJ	N/N	N\N
AUX	(S\NP)/(S\NP)	(S¦NP)/(S¦NP)
ADV_VP	(S\NP)/(S\NP)	(S\NP)¦(S¦NP)
ADV_S	S\S	S¦S
PREP_ADV	((S\NP)/(S\NP))/NP	((S\NP)¦(S\NP))/NP
PREP_N	(N/N)/NP	(N\N)/NP
REL_SUJ	(N/N)/(S\NP)	(N\N)/(S¦NP)
REL_OBJ	(N/N)/(S/NP)	(N\N)/(S¦NP)
VdiT	(S\NP)/NP*NP	(S¦NP)/NP*NP

Ejemplo:



II. GRAMATICAS CATEGORIALES Y UNIFICACION

La aplicación de las Gramáticas Categoriales al ámbito de la Lingüística Computacional ha motivado la adopción de las matrices de rasgos tan usuales en estos sistemas. Efectivamente, trabajos como los de Shieber (1987) y Teorías Gramaticales como las de Bresnan y Kaplan (1982) -LFG-, Gazdar, Klein, Pullum y Sag (1985) -GPSG-. Pollard y Sag (1987) -HPSG-, Zeevat, Klein y Calder (1987) -GCU- , Uszkoreit (1986) -CUG- utilizan matrices de rasgos para la especificación de las categorías.

CATEGORIAS MEDIANTE RASGOS

Una categoría es una función parcial de un conjunto de rasgos a un conjunto de atributos.

Todas las categorías de nuestro sistema se definiran a partir de la combinación de dos categorías básicas. Estas categorías básicas són S (oración) y N (nombre). Por el procedimiento ya explicado en (1) se generarían el resto de categorías complejas.

S y N estan formadas por un conjunto de rasgos con sus respectivos rangos de atributos. Así:

	RASGO	RANGO DE ATRIBUTOS
S =	FIN	+,-
	COMP	+,-
	GER	+,-
	PART	+,-
	PRED	+,-
	BAR	1,2
	TEMP	PRES,PAST,FUT
	ASP	PERF,IMP

	CONC	
	NUM	PLU,SING
	PERS	1,2,3
N =	CONT	+,-
	DEF	+,-
	CAS	NOM,ACC,OBL
	BAR	1,2
	CONC	
	NUM	PLU,SING
	PERS	1,2,3
	GEN	MASC,FEM

Así pues, todas las categorías de nuestra gramática estarán formadas por combinaciones de estas matrices de rasgos con asignaciones diferentes.

DAGS, TEMPLATES Y CATEGORIAS

Los investigadores del campo de la GCU establecen una diferencia entre la categoría que se asigna a una clase de entradas léxicas (la clase de los Verbos transitivos, de los intransitivos, de los nombres comunes) y la que se le asigna a una entrada léxica concreta de una clase.

Así, se denomina 'template' de una categoría a la estructura que se asigna a cualquier entrada que pertenezca a una clase. Esta estructura sólo tiene asignados los atributos que comparten todos los miembros de esa clase, atributos que recibiran las entradas léxicas en el momento en que se les asigne ese 'template' . Después para cada caso sólo faltará especificar los rasgos de lexicón idiosincráticos de cada entrada.

Ejemplo de Template para un verbo transitivo (la información expresada en negrita corresponde al template, el resto sería

el dag. Para ser una entrada léxica aún faltaría instanciar algunas de las variables):

FIN +	CONT xCONT	CONT xCONT
COMP xCOMP	DEF xDEF	DEF xDEF
GER -	CAS NOM	CAS ACC
PART -	BAR 2	BAR 2
PRED -	CONC	CONC
BAR 2	NUM xNUM	NUM x1NUM
CONC	PERS xPERS	PERS x1PERS
		GEN x1GEN

En nuestro sistema de producción de categorías aún hay otra estructura a tener en cuenta. Así es, la primera estructura es el dag (grafo acíclico dirigido) que le asignamos a esa categoría, sin especificar ningún rasgo. El segundo paso es la asignación de valores generales, es decir, la construcción del template. En tercer lugar, la asignación de atributos concretos a una entrada léxica determinada. Eso quiere decir que para hacer una entrada de diccionario necesitamos haber definido previamente un dag y un template de esa categoría, aunque también se podría hacer la entrada léxica directamente en el dag. No obstante, con ello perderíamos la generalidad que proporciona el template.

Un dag destinado a ser un template de una categoría que pertenezca al conjunto de categorías válidas para una Gramática Categorial ha de tener esta forma:

```
DAG_TEMPLATE = RES DIR ARG
RES = C
ARG = C
C = CatBas
C = DAG_TEMPLATE
CatBas = S
CatBas = N
```

El orden en que aparecen los pares atributo valor en el dag es siempre el mismo. Eso quiere decir que los dags usados en nuestro sistema son estructuras rígidas , lo cual hace posible el uso de una unificación tipo *Prolog*.

UNIFICACION

Hemos tomado como modelo de unificación la de tipo *Prolog* ya que es suficientemente potente para tratar los problemas que nos hemos propuesto y, además, su rigidez no supone ninguna limitación. Simplemente obliga al usuario a predefinir el tipo de dags que quiere utilizar. Por ejemplo, si definimos un dag 'N' que aparece dentro de otros dags como NP/**N** o **N****N**, además de en **N** como nombre común, siempre utilizaremos el mismo dag (como estructura sin información) para la generación de los templates que contienen, en este caso, N. Así sabemos que la unificación tipo *Prolog* será siempre adecuada y mantenemos una cierta elegancia en el sistema.

III EL PARSER

LAS REGLAS

Las Reglas de las Gramáticas Categoriales no requieren traducción a Prolog. Quedan como sigue:

Aplicación_hacia_adelante(X/Y , Y , X).

Apli8cación hacia atrás(Y , X\Y , X).

Composición_hacia_adelante(X/Y , Y/Z , X/Z).

Composición hacia atrás(Y\Z, X\Y --><B X\Z.

Y el resto de reglas que quisiera añadirse quedaría expresado análogamente.

Sobre el conjunto de reglas total que se especifique, el usuario puede definir un conjunto de secuencias. Una secuencia puede ser una regla unaria más una regla binaria o una regla binaria. Con ello se acota el rango de reglas que el parser utilizará. Un posible ejemplo de secuencias es:

```
secuencia([Aplic/]).
secuencia([Aplic\]).
secuencia([T> , A>]).
secuencia([T> , C>]).
secuencia([S<]).
```

que es el conjunto que hemos utilizado nosotros.

EL CHART

El parser utiliza una tabla de cadenas bien formadas donde para cadena sencilla (entradas léxicas) o producto de la combinación de otras cadenas se añade un 'arco'.

Por ejemplo, si estamos analizando:

```
0  el 2 niño 3 come 4 manzanas 5
    --------  ---------  ---------  -------------
```

tendremos en el chart arcos tales como:

```
arco(0, 1, el, NP/N).
arco(1, 2, niño, N).
     .........
```

Cada arco indica: inicio, fin, cadena, dag. Para cada combinación válida añadiremos un arco. Así al combinar 'el' con 'niño', añadiremos:

```
arco(0, 2, 'el niño' , NP).
```

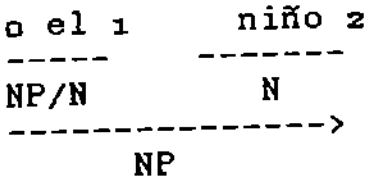
EL ALGORITMO DE COMBINACION

El algoritmo es ascendente y de izquierda a derecha. Toma una lista de elementos y va combinándolos sucesivamente hasta obtener una lista de un sólo elemento.

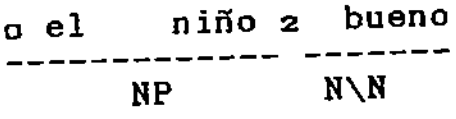
En el caso más sencillo, se combinaría el primer elemento (e_1) con el segundo (e_2) para obtener (e_1 2). A continuación, éste último (e_1 2) se combinaría con el tercero (e_3) para obtener (e_1 2 3) y éste a su vez con el cuarto para formar (e_1 2 3 4) ... Sin embargo, en la mayoría de los casos el parser irá combinando elementos adyacentes entre sí pero no estrictamente de izquierda a derecha.

Al intentar combinar dos cadenas pueden darse tres casos:

1. Que mediante alguna de las reglas escogidas puedan combinarse las dos cadenas:



2. Que no puedan combinarse y la cadena derecha necesite combinarse a la izquierda:



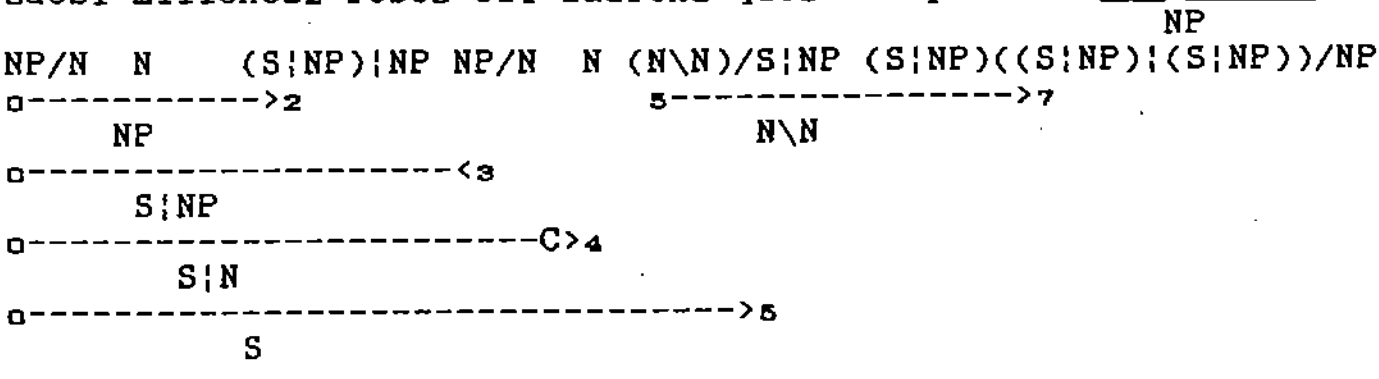
3. Que no puedan combinarse y la cadena derecha no necesite combinarse a la izquierda.

El el primer caso pasamos a combinar la nueva cadena

('el niño') con la cadena que siga. En el segundo, cortamos el desarrollo del programa con un Fail, obligándole a buscar otra forma de combinación para la parte que llevamos analizada. Y en el tercero, dejamos la cadena izquierda (c_i) momentáneamente y pasamos a combinar la cadena derecha (c_d) con la cadena siguiente (c_s), obtenirendo ($c_d\ s$). Luego intentamos combinar (c_i) con ($c_d\ s$) para obtener ($c_i\ s$).

A continuación haremos el seguimiento de un ejemplo, con el que esperamos que quede claro el funcionamiento del analizador.

EJEMPLO

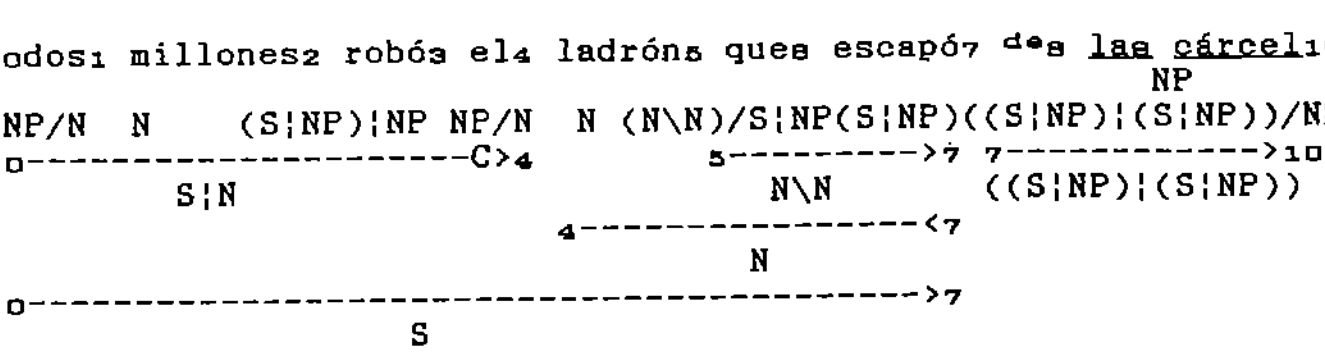


Hasta llegar a la posición 5 el parser sólo se encuentra con el caso 1 de los 3 estados posibles citados en el apartado anterior, por lo que simplemente combina (0,1) con (1,2) y crea (0,2); combina (0,2) y (2,3) con lo que crea (0,3)...

Al llegar a la posición 5 nos encontramos con el estado 3 S<0,8> y (N\N)/(S¦NP)<5,8> no son combinables. Combina (5,6) con (6,7) y intenta combinar (0,5) con (5,7).

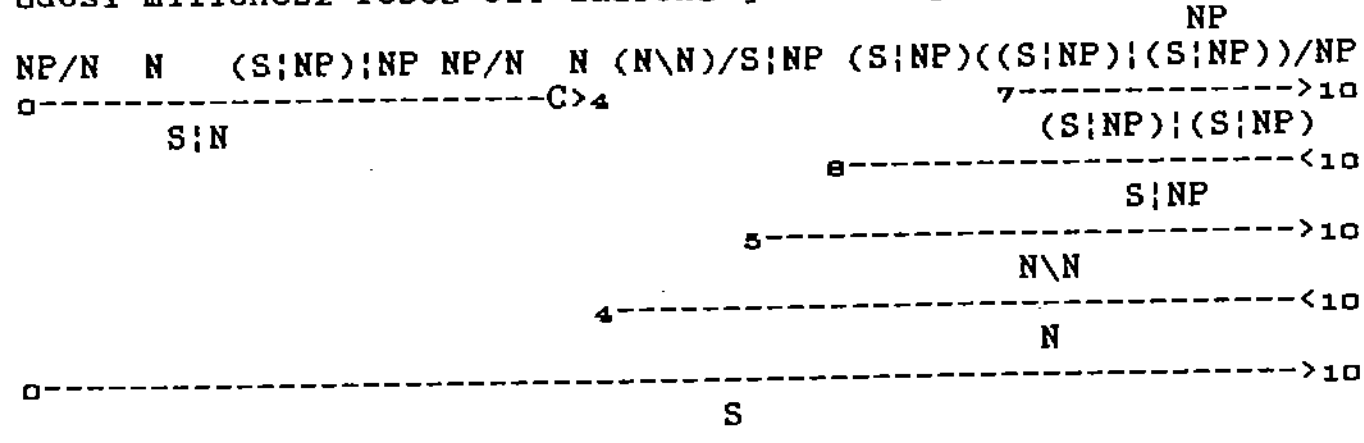
Al intentar combinar (0,5) con (5,7) estamos en el

estado 2 , por lo tanto se fuerza el backtracking. La situación es la siguiente:



El backtracking hace que en lugar de seguir por 1 siga por 3. Al intentar continuar (4,5) con (5,6) nos encontramos otra vez en 3 con lo que intenta combinar (5,6) con (6,7) tomando del chart (5,7), que ya había analizado anteriormente. De esta forma sí puede combinar (5,7) con una cadena a su izquierda y también puede combinar (0,4) con (4,7) creando un arco (0,7).

Con el resto de la oración se producirá una situación similar a la ya explicada:



El parser combinaría hasta obtener una cadena (7,10), y hará backtracking hasta poder combinar (7,10) con (6,7). De

esta forma se llegaría a un análisis sintáctico de una frase que contiene fenómenos lingüísticos controvertidos como son la extracción, inversión de sujeto y una frase de relativo , mostrando de esta forma el interés de las Gramáticas Categoriales para el análisis del lenguaje natural y la naturalidad , elegancia y sencillez de un parser que utilice dichas gramáticas.

BIBLIOGRAFIA

Bach,E. (1988) "Categorial Grammars as theories of Language" dins Oehrle,Bach, Wheeler (eds)

Benthem Van,J. (1988) "The Lambek Calculus" dins Oehrle, Bach, Wheeler (eds)

Benthem,J. (1987) *Categorial Grammar and Type Theory.* Institute for Language,Logic and Information.

Benthem,J. (1989) *The Fine Structure of Categorial Semantics.* Institute for Language,Logic and Information.

Buszkowski,W. (1988) "Generative Power of Categorial Grammars" dins Oehrle, Bach, Wheeler (eds).

Calder;Klein;Moens;Zeevat. (1987) "Problems of Dialogue parsing" . Centre for Cognitive Science. Edinburg.

Calder, Moens, Klein (1987) *Categorial Grammar, Unification Grammar and Parsing.* EWPCS. Vol I. Centre for Cognitive Science.

Calder;Moens;Zeevat. (1986) "A UCG Interpreter" Centre for Cognitive Science. Edinburg.

Dowty,D. (1988) "Type Raising. Functional Composition and non Constituent Conjunction" dins Oehrle, Bach, Wheeler (eds)

Gardent;Jurie;Baschung (1989) "Efficient Parsing for French" Edinburg & Clermont

Haddock,N. (1987) "A Model of Incremental Interpretation" dins Haddock,Klein,Morrill(1987).

Karttunen,L. (1986) "Radical Lexicalism" CSLI.Stanford.

Keenan,E.;Timberlake,A. (1988) "Natural Language Motivations Extending Categorial Grammars" dins Oehrle, Bach, Wheeler (eds)

Lambek,J. (1988) "Categorial and Categorical Grammars" dins Oehrle, Bach, Wheeler (eds)

Linden,E-J. (1989) "Lambek Theorem Proving and Feature Unification" 4th. Conference of the EACL. Manchester.

Moortgart,M. (1989) "Generalized Categorial Grammar: The Lambek-Gentzen Calculus" ms. Edinburg.

Morrill,G. (1989) "Categorial Grammar and Natural Language" ms. Edinburg.

Oehrle,R.T.,Bach,E.;Wheeler,ed (1988) *Categorial Grammars and Natural Language Structures.* D. Reidel Publishing Company.

Pareschi,R. (1987) "Combinatory Grammar,Logic Programming and Natural Language Processing" dins Haddock,Klein,Morrill (1987).

Pareschi,R.;Steedman,M. (1987) "A lazy way to chart-parse Categorial Grammars"

Pollard,C. (1987) "Categorial Grammar and Phrase Structure Grammar: An Excursion on the Syntax Semantics Frontier" dins Oehrle, Bach, Wheeler (eds)

Steedman,M. (1988) "Combinators and Grammars" dins Oehrle, Bach, Wheeler (eds)

Steedman,M. J. (1989) *Work in Progress: Combinators and Grammars in Natural Language Understanding* ms. Edinburg.

Uszkoreit,H. (1986) "Categorial Unification Grammar" COLING.

Whitelock,P.J. (1988) "A feature Based Categorial Morpho-syntax for Japonese." dins Natural Language Parsing and Linguistics

Wood,M.M. (1988) *Categorial Syntax for Coordinate Constructions.* Tesi Doctoral. Londres.

Zeevat,H. (1988) "Combining Categorial Grammar and Unification" dins Natural Language Parsing and Linguistics

Zeevat;Klein;Calder (1987) "Unification Categorial Grammar" dins Haddock,Klein,Morrill(1987).