Aportaciones a la resolución de extraposición de elementos en lenguaje natural utilizando técnicas incrementales

Memoria de Investigación

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante
Carretera San Vicente S/N. 03080 ALICANTE, España.
URL: http://gpl.dlsi.ua.es
patricio@dlsi.ua.es

Presentado por Patricio Manuel Martínez Barco.
Dirigido por Dr. Manuel Palomar Sanz.

Índice

1 Introducción

El *Procesamiento del Lenguaje Natural* se estudia desde distintos puntos de vista en función de los objetivos que se persigan: lexicografía computacional, analizadores morfológicos, reconocimiento automático del habla, análisis sintáctico–semántico, traducción automática, etc. Cada uno de ellos constituye un campo de trabajo con una problemática, una terminología y una tecnología especifica. En particular, en el campo de análisis sintáctico–semántico, la principal línea de investigación se centra en el análisis de oraciones con el fin de proporcionar interfaces amigables en Lenguaje Natural a sistemas basados en el conocimiento [Dahl94a]. El objetivo que se plantea en este análisis es la obtención de la representación de las estructuras sintácticas del lenguaje. Para ello el principal problema que se aborda es cómo conseguir el análisis de infinitas oraciones del lenguaje mediante mecanismos finitos. Sin embargo este objetivo resulta computacionalmente difícil de tratar por lo que generalmente se opta por acotar el lenguaje, restringiéndolo a un subconjunto de oraciones [Moreno93]. Otro problema abordado es el de la obtención de una única representación o estructura sintáctica de una determinada oración ya que el lenguaje natural es ambiguo y las causas y efectos que producen ambigüedad son múltiples [Moreno96].

Cuando se requiere que el sistema de procesamiento automático de un subconjunto del Lenguaje Natural sea más ambicioso, es decir, que la cobertura del lenguaje sea lo suficientemente amplia incluyendo oraciones complejas, entonces surge una serie de problemas adicionales que incrementan notablemente las dificultades de tratamiento [Palomar93][Palomar96]: extraposición, elipsis, anáfora, etc. Algunas aproximaciones gramaticales, las más extendidas, utilizan los formalismos gramaticales lógicos para la construcción automática de estructuras sintácticas y semánticas, proporcionando los mecanismos adecuados para el tratamiento de estos fenómenos. Entre estos formalismos podemos destacar las Gramáticas de Huecos [McCord91], Gramáticas de Unificación de Árboles [Popowich89][Popowich93] y Gramáticas Datalog [Dahl94b][Dahl95].

Este trabajo, en el que hemos desarrollado un planteamiento de nuevos mecanismos para el tratamiento de la extraposición a izquierdas mediante técnicas de análisis ascendente sobre Gramáticas Datalog, se encuentra englobado dentro de la parte de *modelado lingüístico* en el proyecto *Construcción de analizadores híbridos de lenguajes naturales*[1] subvencionado por la Comisión Interministerial de Ciencia y Tecnología. Este proyecto, cuyo objetivo final es la construcción de analizadores definidos sobre modelos lingüísticos y estocásticos, está siendo desarrollado actualmente por el Grupo de Programación Lógica y Sistemas de Información de la Universidad de Alicante junto al Subgrupo de Lenguaje Natural del Grupo de Programación Lógica e Ingeniería del Software y miembros del Grupo de Reconocimiento de Formas e Inteligencia Artificial de la Universidad Politécnica de Valencia.

El método que expondremos para resolver el fenómeno lingüístico de la extraposición está basado en el mecanismo de paralelismo sintáctico–semántico que hemos definido en [Saiz97]. En este método hacemos uso de los programas Datalog Extendidos [Dahl95] [Moreno97] como herramienta para la definición e implementación de gramáticas. Sobre estas gramáticas aplicaremos técnicas de análisis incremental que nos permitirán pasar por distintos niveles de derivación en los cuales podemos congelar el proceso para aplicar restricciones destinadas a la resolución del fenómeno lingüístico. Todo ello está basado en un amplio trabajo de recopilación bibliográfica que hemos reflejado en [Ferrández97].

2 Tratamiento de las cláusulas de relativo.

2.1 El problema.

El problema de las cláusulas de relativo en el lenguaje natural está enmarcado dentro de los fenómenos lingüísticos de *dependencia llenadores–huecos*. Estos fenómenos ocurren cuando un subconjunto de alguna frase (el *hueco* o *traza*) se pierde de su lugar normal a otro sintagma

[1] Proyecto CICYT nº TIC97-0671-C02-01/02

(denominado en algunas ocasiones *llenador*), que se encuentra fuera del sintagma incompleto, representando al componente perdido [Pereira87]. La ocurrencia de un *hueco* se dice que está licenciada por la ocurrencia previa del *llenador*, y existirá una dependencia entre el *hueco* y el *llenador* debido a que el hueco sólo puede darse cuando aparece el *llenador* adecuado. El fenómeno de la dependencia de *llenadores–huecos* ocurre típicamente en lenguas como el inglés o el castellano en las cláusulas de relativo y en cláusulas interrogativas.

Las *dependencias entre los llenadores–huecos* son una subclase de *las dependencias de larga distancia* o *dependencias ilimitadas*. Éstas se llaman así por la abundancia de información que se puede encontrar entre las clases dependientes (el *hueco* y el *llenador*, en este caso), y porque el camino entre ambos en el árbol de análisis puede cruzar varias fronteras de frase (aunque sólo ocurre con cierto tipo de frases).

Para resolver este tipo de fenómenos se requiere el uso de gramáticas que expresen no sólo información sintáctica de cada componente, sino también las relaciones sintácticas y semánticas que puedan tener éstos con otros constituyentes.

En concreto, en las *cláusulas de relativo* se produce un efecto de *dependencia llenador–hueco* conocido como *extraposición a izquierdas*, que tal y como se define en [Pereira87] "ocurre en una oración cuando un subconstituyente de un constituyente que forma parte de la oración, se representa por otro a la izquierda del que está incompleto".

2.2 La oración de relativo. Sus tipos.

Las oraciones de relativo (conocidas también como proposiciones adjetivas) cumplen el efecto de complementar siempre a un *sintagma nominal* al que llamamos *antecedente*. En castellano, las *oraciones de relativo* se introducen mediante los siguientes pronombres [Lázaro85]:

que
quien (quienes)
el cual (la cual, los cuales, las cuales)
cuyo (cuya, cuyos, cuyas)

de los cuales se conoce que

a) El pronombre *que* reproduce antecedentes con rasgos humanos, animales o de objeto.
b) El antecedente del pronombre *quien* debe poseer necesariamente rasgo humano. Además usa el mismo rasgo morfológico de número que tiene su antecedente.
c) *Cual* se comporta de la misma manera que *que* con la diferenciación de que usa los rasgos morfológicos de género y número que tiene su antecedente.
d) *Cuyo* tiene carácter posesivo y hereda los rasgos morfológicos del elemento al que determina.
e) En muchas ocasiones se sustituye el uso de *cuyo* por el de *que su*, por ejemplo, "*He conocido a la chica que su padre es notario*" en lugar de "*He conocido a la chica cuyo padre es notario*"

El razonamiento humano utiliza esta información para resolver problemas de ambigüedad. El sistema que se presenta aquí también aprovecha estos condicionantes que nos impone el lenguaje, si bien, aunque el tratamiento es paralelo, para simplificar la exposición nos centraremos sólo en el procesamiento de las oraciones de relativo introducidas mediante el pronombre relativo *que* consideradas como las más representativas.

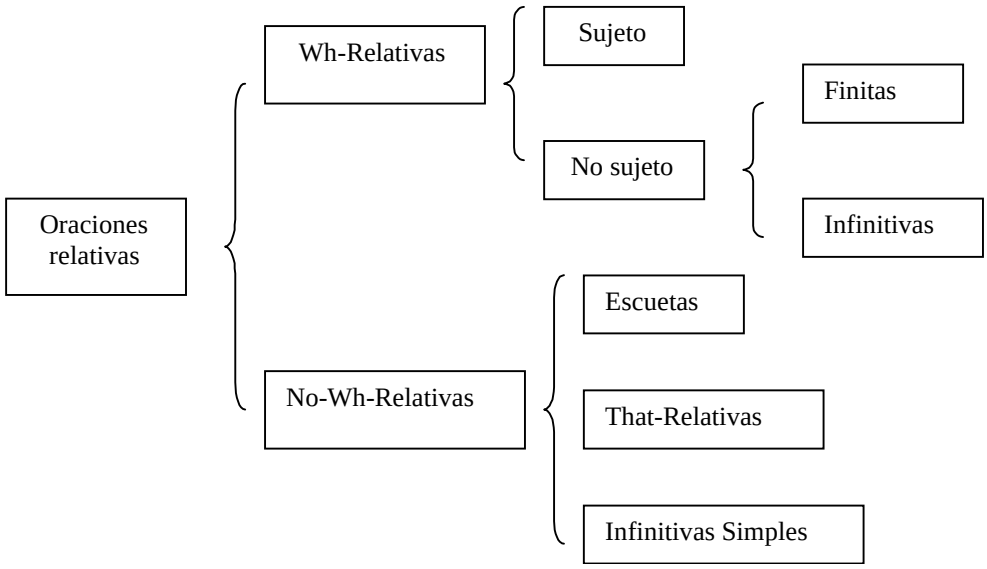
En el caso concreto del pronombre relativo *que* sabemos que se introduce en las oraciones evidenciando la ausencia de algún sintagma nominal, de tal forma que atendiendo a la función sintáctica que adopte éste se originan los siguientes tipos de oraciones de relativo [Callejo97] en castellano:

CLASIFICACIÓN DE ORACIONES DE RELATIVO SEGÚN EL ELEMENTO EXTRAPUESTO

Sujeto	El empleado que me atendió era muy amable.
Complemento directo	Eso que dices no es cierto.
Atributo	Por muy barato que sea ese coche no lo compro.
Suplemento	Este es el libro de que te hablé.
Complemento indirecto	Los asuntos a que te dedicas no son muy legales.
Complemento circunstancial	El pueblo en que nací es pequeño.
Adyacente preposicional	Prestadme toda la atención de que seáis capaces.

Nótese que las cuatro últimas se introducen mediante la ayuda de una preposición. Esto las hace fácilmente distinguibles.

En [Sag97] se hace un estudio exhaustivo de la diversidad de oraciones de relativo en inglés del que se ha extraído la siguiente clasificación atendiendo a las restricciones que se plantean en cada uno de los casos:



Algunos ejemplos de esta clasificación los tenemos en:

a) Wh–Relativas de sujeto:

The man *who visited Kim*.

The man *whose mother visited Kim*.

b) Wh–Relativas de no–sujeto finitas:

The man *who Kim visited*.

The man *whose mother Kim visited*.

c) Wh–Relativas de no–sujeto infinitivas:

Introducidas por el complemento infinitivo del inglés *to*:

The man in *whom to place your trust* ...

d) That–Relativas:

The house *that you visited* was built in 1909.

e) No–Wh–Relativas escuetas:

Oraciones relativas con pronombre relativo elidido, como en:

The house (that) *you visited* was built in 1909.

f) Infinitivas simples:

Introducidas por el *to* infinitivo sin pronombre relativo:

The problems *to solve* are the ones in the Times.

Pensando en restricciones, las oraciones *That–Relativas* siguen la misma clasificación que las *Wh–Relativas* puesto que el pronombre *that* tiene el mismo tratamiento que *who* sólo que este último tiene el rasgo semántico de la animación.

Para nuestro trabajo vamos a aplicar la clasificación de I. Sag [Sag97] al castellano con lo cual obtenemos dos efectos interesantes, por un lado, tenemos una clasificación por restricciones (especialmente útil para nuestro sistema basado en restricciones) y por otro lado, tenemos una clasificación generalizada que nos permite su aplicación tanto al castellano como al inglés (una lengua romance y una germánica).

Así, centrándonos en el caso de las oraciones introducidas por el pronombre relativo *que*, distinguiremos entre las "*oraciones de relativo de sujeto*" y *las* "*oraciones de relativo de complemento verbal*" (Complemento directo, indirecto, circunstancial, atributo, suplemento y adyacente preposicional) puesto que la detección de estas últimas utiliza un mismo mecanismo basado en las características semánticas del verbo.

2.3 Motivación

Analizados los distintos casos posibles, hemos centrado nuestro estudio en el problema de la extraposición a izquierdas en oraciones de relativo introducidas por el pronombre *que*. La decisión de adoptar este caso y no otros es debido a que, por un lado, la resolución de la extraposición a izquierdas es un caso concreto dentro de los fenómenos llamados de llenador–hueco. Si conseguimos resolver el problema de la extraposición, la resolución de otro tipo de fenómenos de llenador–hueco es inmediata con sólo modificar las restricciones introducidas. Por otra parte, el pronombre *que* es el caso más general en cuanto a los pronombres relativos ya que no contiene restricciones semánticas específicas. La resolución de cualquier extraposición provocada por otros pronombres relativos se simplifica al incorporar restricciones semánticas que ayudarán en la identificación del elemento extrapuesto (como ocurre en quien, quienes, cuyo, etc.). Por ejemplo, en la frase "*El hombre de la cazadora gris quien salió a recibirte ...*" el pronombre nos proporciona a priori información semántica para resolver el elemento extrapuesto sobre el sintagma nominal cuya cabecera es *hombre* y no en el que tiene *cazadora* como cabecera puesto que, por definición, *quien* debe tener un antecedente humano. Si esta misma frase la reconstruimos con el pronombre *que*: "*El hombre de la cazadora gris que salió a recibirte ...*" será necesario recurrir al análisis de los rasgos semánticos del verbo *salir* para identificar el antecedente.

Por último, la resolución de problemas de extraposición mediante técnicas de concordancia de rasgos semánticos nos crea una base para la resolución de otros fenómenos lingüísticos.

Es importante destacar que si bien el caso de las oraciones de relativo en inglés no serán objeto de nuestro estudio, el tratamiento de las mismas se realizará de forma paralela aunque añadiendo pequeñas modificaciones a las restricciones.

3 Antecedentes

> *Este apartado está basado en el trabajo: "Resolución de la extraposición a izquierdas con las Gramáticas de Unificación de Huecos", Procesamiento del Lenguaje Natural nº 21." [Ferrández97]*

Una de las primeras aportaciones a la resolución del problema de la extraposición se proporciona en las gramáticas de metamorfosis (MG) de A. Colmerauer [Colmerauer78]. Se trata de reglas de reescritura donde los símbolos no terminales pueden tener argumentos y la aplicación de una regla puede llevar a la unificación. Las MG constituyen el primer formalismo gramatical que usa términos lógicos para representar símbolos gramaticales.

Las reglas de una gramática de metamorfosis tienen como condición que la parte izquierda debe empezar siempre por un símbolo no terminal, pudiendo estar seguido de una secuencia de terminales. A su vez, la parte derecha puede contener cualquier secuencia de terminales y no–terminales:

s1, [s2], [s3], [s4] → [s2], [s3], s1, [s4]

En estas gramáticas, los símbolos [ ] delimitan terminales.

Posteriormente, Pereira y Warren [Pereira80] definen las Gramáticas de Cláusulas Definidas (DCG) como una restricción de las MGs. Si una MG es una gramática tipo–0 restringida donde se usan los términos lógicos para representar símbolos gramaticales, una DCG es una Gramática Libre de Contexto (CFG) donde se usan términos lógicos para representar símbolos gramaticales.

Las DCG sólo permiten en la parte izquierda de la regla un no–terminal simple, como:

sintagma_verbal → verbo(x,y), objeto(y)

Una gramática DCG está basada en reglas CFG como las siguientes [Pereira81]:

CFG para oraciones de relativo

sentencia → sintagma_nominal, sintagma_verbal.
sintagma_nominal → nombre_propio.
sintagma_nominal → determinante, nombre, relativo.
sintagma_nominal → determinante, nombre, sintagma_preposicional.
sintagma_nominal → traza.
traza → [ ].
sintagma_verbal → verbo, sintagma_nominal.
sintagma_verbal → verbo.
relativo → [ ].
relativo → pronombre_relativo, sentencia.
sintagma_preposicional → preposición, sintagma_nominal.

En las DCG es necesario refinar estas reglas para arrastrar los constituyentes extrapuestos mediante la adición de una serie de argumentos a los símbolos no terminales. Además, en el caso de la regla *sintagma_nominal → traza* se permite expandir un sintagma nominal en una traza aunque nos encontremos fuera de una cláusula relativa. Para evitar esto se introducen argumentos que indicarán si la frase en análisis puede estar extrapuesta. A continuación se muestra la gramática DCG resultante:

DCG para oraciones de relativo

sentencia_completa → sentencia(nil).

sentencia(Hueco0) → sintagma_nominal(Hueco0,Hueco1), sintagma_verbal(Hueco1).

sintagma_nominal(Hueco, Hueco) → nombre_propio.

sintagma_nominal(Hueco,Hueco) → determinante, nombre, relativo.

sintagma_nominal(Hueco0,Hueco)

→ determinante, nombre, sintagma_preposicional(Hueco0, Hueco).

sintagma_nominal(traza,nil) → traza.

traza → [ ].

sintagma_verbal(Hueco) → verbo, sintagma_nominal(Hueco, nil).

sintagma_verbal(nil) → verbo.

relativo → [ ].

relativo → pronombre_relativo, sentencia(traza).

sintagma_preposicional(Hueco0,Hueco) → preposición, sintagma_nominal(Hueco0,Hueco).

Puesto que las DCG tienen como condición que la parte izquierda de la regla sólo puede contener un único símbolo terminal (por CFG), se hace necesario un número excesivo de reglas para representar el concepto de la extraposición.

El formalismo de Colmenauer de las MG permite una forma alternativa de expresar la extraposición izquierda, usando reglas cuya parte izquierda es un no–terminal seguido de una cadena de símbolos terminales “tontos” que no tienen ocurrencia en el vocabulario de entrada. Como ocurre, por ejemplo, en:

marcador_relativo, [t] → pronombre_relativo

que indica que un pronombre relativo puede analizarse como un marcador relativo haciendo que el terminal t se añada al principio de la entrada restante después de la aplicación de la regla.

Aunque éste es un método que resuelve el problema de las DCG, cuando la gramática se hace extensa sufre de los mismos problemas de claridad que éstas. Además, contiene otro problema: en general, el lenguaje que define una gramática debe contener sentencias extra para representar los terminales tontos. Sin embargo, el análisis de estos terminales no ha de suponer ningún problema al analizador por lo que podrían ser omitidos simplificando la gramática.

Las DCG proporcionan una maquinaria básica para una clara descripción de los lenguajes y sus estructuras. Sin embargo, carece de mecanismos para describir de forma simple el problema de la extraposición y sus restricciones asociadas.

Las MG pueden expresar la reescritura de múltiples símbolos en una regla simple, pero los símbolos deben ser contiguos, como en una regla de gramática tipo–0. La descripción del problema de la extraposición izquierda supone complicar el resto de la gramática en exceso.

Después de analizar todos estos inconvenientes, F. Pereira [Pereira81] propone las Gramáticas de Extraposición (XG) con una nueva característica: Los saltos entremezclados en la parte izquierda de la regla serán rutinariamente reescritos en orden secuencial en la parte más a la derecha de la regla:

marcador_relativo, salto (x), traza $\rightarrow$ *pronombre_relativo, salto(x).*

En la parte izquierda de la regla hay un constituyente vacío, la *traza*, que ocupa el hueco que ha dejado un constituyente que se ha perdido. El *marcador* indica que un constituyente a su derecha contiene una traza. Se puede ver como que el constituyente, en cuyo lugar está ahora la traza, ha sido extrapuesto a la izquierda, y su nueva posición se representa con el marcador.

Una XG válida podría ser la siguiente:

sentencia → sintagma_nominal, sintagma_verbal.

sintagma_nominal → determinante, nombre, relativo.

sintagma_nominal → traza.

relativo → [ ].

relativo → marcador_relativo, sentencia.

marcador_relativo ... traza → pronombre_relativo.

Todas las reglas, a excepción de la última, son libres de contexto. La última representa la extraposición como una cláusula relativa simple.

La diferencia entre las reglas XG y las reglas DCG está en que la parte izquierda de una regla XG puede contener varios símbolos. Así permite la expansión simultánea de algunos símbolos no contiguos en una cadena. Por otra parte, la XG cumple la misma propiedad fundamental que una DCG: se trata de una notación apropiada para representar cláusulas definidas en un programa lógico ordinario.

Se puede concluir que las XG describen el fenómeno de la extraposición izquierda con potencia por los siguientes motivos:

- Son una extensión de las DCG que pueden ser interpretadas como formalismos gramaticales independientemente de su traducción a cláusulas definidas.
- Proporcionan una descripción simple de la extraposición izquierda y de sus restricciones: las restricciones de la extraposición pueden expresarse usando algunas herramientas artificiales como las cadenas de encorchetado, es decir, cadenas de inicio y fin que encierran un subcomponente.
- Se pueden comparar en eficiencia con las DCG cuando se ejecutan en PROLOG.

Por contra, la introducción de restricciones por medio del encorchetado de las frases extrapuestas en las XG de Pereira generan gramáticas y derivaciones complejas y de difícil comprensión. Además, tal y como apunta el mismo Pereira, la conexión entre el formalismo de las XG y las estrategias de análisis tradicionales no es una tarea trivial, lo que complica su implementación.

V. Dahl [Dahl89] propone una generalización de las XG conocida como las Gramáticas Discontinuas (DG) (llamadas primitivamente Gramáticas de Huecos). Estas gramáticas fueron implementadas por V. Dahl y M. McCord [Dahl83] aplicándolas a la resolución del problema del tratamiento de la coordinación en un compilador conocido como SYNAL. V. Dahl junto a H. Abramson [Dahl84a], y posteriormente F. Popowich [Popowich86], hacen un estudio de conclusiones sobre este compilador. El SYNAL es incrementado por Dahl y Saint–Dizier [Dahl86] añadiéndole la característica de los mecanismos de restricción con el propósito de automatizar las restricciones lingüísticas.

V.Dahl [Dahl84b] generaliza el uso de las DG aplicándolas a otros problemas del procesamiento del lenguaje entre los que se encuentra el problema de la extraposición izquierda en las oraciones de relativo. Las DG son una generalización de las XG donde los símbolos no identificados pueden ser repuestos arbitrariamente (sin necesidad de hacerlo al final de la cadena como ocurre en XG). Una discontinuidad o salto es una subcadena que se separa y se repone sin analizarse, para ser analizada en su nueva localización.

Las DG añaden las siguientes características:

- Permite una reposición libre de las discontinuidades: En el ejemplo de sintagma nominal "*el hombre con cuyo tío Juan partió*" se produce una doble extraposición izquierda que sería difícil de expresar en XG: "*el hombre [Juan partió con [el] tío [del hombre]]*". En las DG una regla simple podría capturarlas:

$$np(x), salto(y), prep, det, salto(z), prep(de), np(x) \rightarrow np(x), prep, [cuyo], salto(z), salto(y)$$

- Permite otras formulaciones discontinuas equivalentes: Los saltos pueden reescribirse al final de la parte derecha o en cualquier otro orden cambiando el sentido de la gramática y aceptar los mismos lenguajes obteniendo, en algunos casos, mejores costes.
- Permite interaccionar entre diferentes reglas de discontinuidad: Las XG sólo permitían múltiples saltos si éstos eran independientes o estaban anidados totalmente uno dentro de otro. Las DG permiten cualquier tipo de salto aunque interaccionen entre sí.
- No necesita los símbolos artificiales de las cadenas encorchetadas para expresar restricciones.
- Obtiene gramáticas mas simples sin perder su eficiencia.
- Permite tratar lenguajes naturales donde las palabras siguen una ordenación libre de palabras (atendiendo más a su declinación que al orden establecido, como el Latín o el Sánscrito).
- Trata eficientemente el problema de la libre ordenación de constituyentes que pueden o no estar presentes dependiendo de otros constituyentes. Es el caso de los verbos que requieren sólo un objeto directo mientras otros requieren también un indirecto.

Las DG se quedan, sin embargo, en el tratamiento sintáctico de la extraposición, pero no tratan de forma clara el problema de las restricciones en este movimiento. V. Dahl tratará posteriormente un subconjunto de las DG, las Gramáticas Discontinuas Estáticas (SDG) [Dahl88], en las que sólo se mueven los constituyentes definidos sobre las discontinuidades mientras los saltos permanecen fijos. Estas SDG proporcionan el filtro necesario para el tratamiento de las restricciones del lenguaje que propone Chomsky en su Teoría de Rección y Ligamiento [Chomsky82][Chomsky88].

Otros formalismos han tratado el problema de la extraposición basándose en gramáticas no concatenantes, como las Gramáticas de Cabeceras (HG) de Pollard [Pollard84] y múltiples derivaciones de las Gramáticas de Árboles Contiguos (TAG) de Kroch y Joshi [Kroch86]. En concreto, las HG se basan en el principio de que la cabecera de un constituyente es el elemento clave de este constituyente. El formalismo de las HG divide un constituyente en tres componentes: un contexto izquierdo, un terminal cabecera y un contexto derecho. En la

reescritura de una regla HG, las tres partes de un constituyente pueden colocarse de distinta forma a la de su construcción. Estas gramáticas tienen como inconveniente que para cada tipo de constituyente hay una cabecera única, es decir, un elemento que se puede mover. Por esto no sería posible tratar varios problemas de extraposición dentro de un mismo constituyente de forma simultánea.

Recientemente, Groenink [Groenink95] define las Gramáticas de Movimiento de Literales (LMG) como solución al fenómeno de la extraposición a través de un mecanismo que permite un desplazamiento top–down de información sintáctica, basándose en las gramáticas HG, XG y TAG que ya han usado métodos de desplazamiento sintáctico.

Las LMG separan el tratamiento del lenguaje natural en dos fases: una fase de análisis según el tratamiento tradicional libre de contexto y una segunda fase de eliminación de ambigüedad mediante matching, a diferencia de métodos anteriores que lo hacen en una fase única. Así, las LMG proporcionan una forma clara y sencilla de tratar el fenómeno de la extraposición siendo capaz de modelar varios movimientos de una vez.

Sin embargo, como indica Groenink, no se han estudiado los rendimientos de las LMG en grandes gramáticas. Sería necesario hacer pruebas en gramáticas completas para determinar exactamente la eficiencia del método.

En cualquier caso, los formalismos anteriores tratan, de forma mas o menos clara, el problema sintáctico de la extraposición pero en ningún caso se trata el problema del análisis semántico. El método que proponemos en este trabajo nos permitirá hacer una evaluación semántica posterior.

4 Gramáticas Datalog Extendidas

Las Gramáticas Datalog (DLG) [Dahl94b] presentadas por Dahl, Tarau y Huang, y extendidas por Dahl, Tarau, Moreno y Palomar [Dahl95] [Moreno97] son una subclase de las Gramáticas de Cláusulas Definidas (DCG) [Pereira80], que usan un predicado de conexión sin símbolos

de función y un conjunto de hechos descritos por representación asercional de la cadena de entrada.

Las DLG presentan grandes ventajas respecto a las DCG: una semántica más simple, un mejor coste computacional, y por último, una mayor eficiencia de implementación al evitar el uso de listas como ocurre en las DCG.

Además, por ser una variante de las DCG presentan las mismas ventajas que éstas respecto a las Gramáticas Libres de Contexto. Así, permiten expresar las dependencias del contexto, permiten la construcción de estructuras de árbol arbitrarias en el proceso de análisis sin verse restringidas por la estructura recursiva de la gramática, permiten incluir condiciones extra en las reglas de la gramática, y extienden las CFG aumentando los símbolos no terminales con nuevos argumentos. Por estos motivos consideramos que las DLG son adecuadas para describir el lenguaje natural.

A continuación presentaremos la formalización de los programas Datalog como una herramienta para la definición e implementación de Gramáticas Libres de Contexto, para posteriormente definir los programas Datalog Extendidos que nos permitirán incluir símbolos de función en los argumentos de los predicados para obtener la representación sintáctica y semántica de las frases de entrada. Finalmente, definiremos el proceso de evaluación incremental para los programas Datalog extendidos mediante el algoritmo *Semi–naive*.

4.1 Programas y Gramáticas Datalog.

Según se introduce en [Dahl94b] con la revisión posterior de [Moreno97], definimos un programa Datalog utilizando el siguiente formalismo:

Definición 1. Sea *L(A, F)* un lenguaje de primer orden, siendo *A* el alfabeto sin símbolos de función, y *F* el conjunto de fórmulas bien formadas; definimos:

Un **programa Datalog** es un conjunto de reglas de la forma:

$A \leftarrow A_1, A_2,\ldots, A_n$;donde A_i es un átomo.

Se pueden distinguir dos tipos de reglas:
hechos: $A \leftarrow$
reglas: $A \leftarrow A_1, A_2,\ldots, A_n$

Un **objetivo Datalog** es una fórmula de la forma:

$\leftarrow A_1, A_2,\ldots, A_n$;donde A_i es un átomo.

Definición 2. La semántica de un programa Datalog viene definida por su mínimo modelo de Herbrand (semántica declarativa)[2].

Definición 3. Dada una Gramática Libre de Contexto *G* con *S* como símbolo inicial y una sentencia de entrada *I*, de la forma:

$I = w_1\ w_2 \ldots w_m$

se define:

Un **programa Datalog asociado a la Gramática Libre de Contexto y a la sentencia *I***, que denotamos como *DL(G,I)* como sigue:

La sentencia de entrada $I = w_1\ w_2 \ldots w_m$ se descompondrá en hechos con sus posiciones, originando las cláusulas *'D'*$(w_1,0,1)$, *'D'*$(w_2,1,2),\ldots$, *'D'*$(w_m, m\text{-}1, m)$, donde *'D'* es un predicado básico que utilizamos para representar los hechos.

[2] Se trata del conjunto de átomos básicos que son consecuencia lógica del programa, es decir, aquello que se puede derivar de él usando las reglas *Datalog*.

La regla léxica $P \rightarrow palabra$ origina la cláusula*: P(A,B) ← 'D'(palabra,A,B).*
Una regla de producción de la forma $P \rightarrow P_1, P_2,\ldots,P_n$ origina la *cláusula* $P(A_0, A_n) \leftarrow P_1,P_2,\ldots,P_n$, donde

$$\begin{cases} P'_i = P_i(A_{i-1}, A_i) & \text{si } P_i \text{ es un símbolo no terminal} \\ P'_i = \text{'D'}(P_i, A_{i-1}, A_i) & \text{si } P_i \text{ es un símbolo terminal} \end{cases}$$

Un **objetivo Datalog asociado a *DL(G, I)*** que denotamos *DL(S, I)* como:
$\leftarrow s\ (0, m).$

Teorema 1. *Sea G una Gramática Libre de Contexto con S como símbolo inicial e I una entrada; entonces existe un programa Datalog asociado a dicha Gramática Libre de Contexto que denotamos DL(G, I) y un objetivo DL (S, I) que tendrá éxito si y sólo si I es reconocida por G.*

Prueba. La prueba es inmediata por la definición 3.

Definición 4. Una **Gramática Datalog** es un *programa Datalog* obtenido por la definición 3.

4.2 Programas y Gramáticas Datalog Extendidas

Como hemos visto anteriormente, los programas Datalog permiten la definición e implementación de Gramáticas Libres de Contexto. Sin embargo, para la resolución de los fenómenos linguísticos necesitamos incluir en la gramática cierta información dependiente del contexto como puede ser, por ejemplo, la representación sintáctica y semántica. Para ello, se extienden los programas Datalog incluyendo símbolos de función como argumentos en sus predicados tal y como se presenta en [Dahl95] y se formaliza en [Moreno97] permitiendo una correcta definición e implementación de las Gramáticas de Cláusulas Definidas.

Definición 5. Sea L(A, F) un lenguaje de primer orden, siendo A el alfabeto de símbolos (con la inclusión de símbolos de función), y F el conjunto de fórmulas bien formadas; entonces definimos:

Un programa Datalog extendido como un conjunto de reglas de la forma:

Un **programa Datalog** es un conjunto de reglas de la forma:

$A \leftarrow A_1, A_2,\ldots, A_n$;donde A_i es un átomo.

Se pueden distinguir dos tipos de reglas:

hechos: $A \leftarrow$
reglas: $A \leftarrow A_1, A_2,\ldots, A_n$

Un **objetivo Datalog** es una fórmula de la forma:

$\leftarrow A_1, A_2,\ldots, A_n$;donde A_i es un átomo.

Definición 6. La **semántica de un programa Datalog extendido** viene definida por su mínimo modelo de Herbrand (semántica declarativa).

Definición 7. Dada una gramática de cláusulas definidas G con símbolo inicial S, y una sentencia de entrada *I* de la forma $I = w_1\ w_2\ \ldots\ w_m$; entonces definimos:

Un **programa Datalog extendido asociado a dicha gramática y a la sentencia *I***, que denotamos como *DLE(G,I)* como sigue:

La sentencia de entrada $I = w_1\ w_2\ \ldots\ w_m$ se descompondrá en hechos con sus posiciones, originando las cláusulas *'D'*$(w_1,0,1)$, *'D'*$(w_2,1,2),\ldots$, *'D'*$(w_m, m\text{-}1, m)$, donde *'D'* es un predicado básico que utilizamos para representar los hechos.

La regla léxica $P(cat,est,ras) \rightarrow palabra$ origina la cláusula*:* $P(cat,est,ras,A,B) \leftarrow$ *'D'(palabra,A,B),* donde *cat*, *est* y *ras* son términos funcionales (construidos con símbolos de función9, que representan la estructura semántica, la estructura sintáctica y los rasgos semánticos asociados a *palabra*, respectivamente.

Una regla de producción de la forma $P(cat,est,ras) \rightarrow P_1(cat_1,est_1,ras_1)$, $P_2(cat_2,est_2,ras_2)$, …,P_n (cat_n,est_n,ras_n), origina la *cláusula:* $P(cat,est,ras,A_0, A_n) \leftarrow P_1,P_2,\ldots,P_n$, donde

$$\begin{cases} P'_i=P_i(cat_i,est_i,ras_i,A_{i-1}, A_i) & \text{si } P_i \text{ es un símbolo no terminal} \\ P'_i='D'(P_i,A_{i-1},A_i) & \text{si } P_i \text{ es un hecho} \end{cases}$$

Un **objetivo Datalog extendido asociado a *DLE(G, I)*** que denotamos *DLE(S, I)* como:
$\leftarrow s\ (0, m)$.

Teorema 2. *Sea G una gramática de cláusulas definidas con S como símbolo inicial e I una entrada; entonces existe un programa Datalog asociado a dicha gramática de cláusulas definidas que denotamos DL(G, I) y un objetivo DL (S, I) que tendrá éxito si y sólo si I es reconocida por G.*

Prueba. La prueba es inmediata por la definición 7.

Definición 8. Una Gramática Datalog Extendida es un programa Datalog extendido obtenido por la definición 7.

4.3 Técnicas incrementales aplicadas a Gramáticas Datalog Extendidas.

Las técnicas incrementales de evaluación se han desarrollado fundamentalmente en el campo de las bases de datos deductivas como una aplicación de los programas Datalog. Las técnicas incrementales de evaluación están basadas en el algoritmo *Semi–naive*.

El algoritmo *Semi–naive* es un procedimiento que permite la obtención del mínimo modelo de Herbrand proporcionando la semántica de una base de datos lógica. En este algoritmo, se

parte de un conjunto de axiomas que, mediante las reglas de derivación, proporcionan los teoremas del primer nivel, y tomando como partida los teoremas obtenidos, se deriva un segundo nivel, y así sucesivamente. Como condición, para la derivación de los teoremas de un nivel es necesario usar, al menos, un teorema derivado del nivel anterior.

Una definición clásica de esta técnica se puede encontrar en [Ullman89]. Posteriormente, en [Huang94] se presenta una versión modificada que incorpora el algoritmo *Semi–naive* aumentado con contadores permitiendo conocer el número de veces que se ha derivado un determinado teorema. Esta última está diseñada para su aplicación al caso particular de la actualización de las Bases de Datos. Es en [Palomar96] donde se aplica por primera vez al caso particular de las Gramáticas Datalog.

Las técnicas de evaluación incremental han sido modificadas en [Moreno97] para su aplicación a las Gramáticas Datalog Extendidas incluyendo símbolos de función como argumentos en los predicados de los programas Datalog extendidos. Este algoritmo es el que vamos a mostrar a continuación; posteriormente lo adaptaremos al tratamiento de los elementos extrapuestos.

Dada una relación P 6–aria, donde los dos primeros campos se refieren a la representación semántica y sintáctica respectivamente, el tercero a los rasgos semánticos, el cuarto y quinto a las posiciones anteriores y posteriores de la palabra, componente o grupo de componentes, y el último al número del contador de prueba, se define:

Algoritmo de generación de datos.

Entrada: P: Hechos base.
Salida: P: Hechos derivados.
Método:

Inicio
$\Delta P := P;$
$P := \emptyset;$

mientras $\Delta P \neq \varnothing$ hacer

$\Delta P_{pivot} := (\Delta P \text{ o } P) \cup (P \text{ o } \Delta P) \cup (\Delta P \text{ o } \Delta P)$;

$P := P \cup \Delta P$;

fin mientras

Fin;

Donde la operación o se define como:

Sean $R_1(cat_1, est_1, ras_1, X_1, X_2, Contador_1)$ y

$R_2(cat_2, est_2, ras_2, X_3, X_4, Contador_2)$ dos relaciones sextarias,

$$R_1 o R_2 = \prod_{cat_1+cat_2,\ res_1+res_2,\ ras_1+ras_2,\ X_1,\ X_4,\ Max(Contador_1,Contador_2)+1} \left(R_1 \underset{R1.X2=R2.X3}{\bowtie} R_2\right)$$

5 Técnica de análisis: El análisis incremental.

A continuación vamos a mostrar un ejemplo que ilustra, partiendo de una gramática DCG, la construcción del programa Datalog asociado (por la definición 7) y sobre éste aplicamos las técnicas de análisis incrementales mediante el algoritmo Semi–naive.

Consideremos la siguiente gramática DCG:

s(s(Sem, gn(G,N),R) → sn(sn(X,Y,Sem), gn(G,N),R), sv(sv(X,Y), gn(G,N), R1)

sn(sn(X,Z,Sem),gn(G,N),R) → det(det(X,Y,Z,Sem), gn(G,N),_), n(n(X,Y),gn(G,N),R)

sv(sv(X,Sem),gn(G,N),R) → v(v(X,Y,S),gn(G,N),R), sn(sn(Y,S,Sem)),gn(G,N),R1)

det(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_) → [el]

v(v(X,Y,construir(X,Y),gn(_,s),(X:[humano],Y:[vivienda])) → [construyó]

n(n(X,arquitecto(X),gn(m,s),[humano]) → [arquitecto]

n(n(X,edificio(X),gn(m,s),[construcción]) → [edificio]

Aplicando la definición 7 obtenemos el siguiente programa Datalog extendido:

s(s(Sem),gn(G,N),R,A,B) ← sn(sn(X,Y,Sem),gn(G,N),R,A,C),sv(sv(X,Y),gn(G,N),R1,C,B)

sn(sn(X,Z,Sem),gn(G,N),R,A,B) ← det(det(X,Y,Z,Sem), gn(G,N),_ ,A,C), n(n(X,Y),gn(G,N),R,C,B)

sv(sv(X,Sem),gn(G,N),R,A,B) ← v(v(X,Y,S),gn(G,N),R,A,C), sn(sn(Y,S,Sem)),gn(G,N),R1,C,B)

det(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,A,B) ← 'D'(el,A,B)

v(v(X,Y,construir(X,Y),gn(_,s),(X:[humano],Y:[vivienda]),A,B) ← 'D'(construyó,A,B)

n(n(X,arquitecto(X),gn(m,s),[humano],A,B) ← 'D'(arquitectoA,B)

n(n(X,edificio(X),gn(m,s),[construcción]) ← 'D'(edificio,A,B)

Tomamos la frase ejemplo de entrada con su representación asercional:

$_0$El $_1$ arquitecto $_2$ construyó $_3$ el $_4$ edificio $_5$

Para cada palabra obtenemos una cláusula de tres argumentos con la palabra, su posición anterior y su posición posterior:

'D'(el,0,1)
'D'(arquitecto,1,2)
'D'(construyó,2,3)
'D'(el,3,4)
'D'(edificio,4,5)

Con el programa Datalog extendido asociado a la gramática y con la frase de entrada intentaremos encontrar una frase entre las posiciones 0 y 5.

Mediante la evaluación incremental del algoritmo Semi–naive vamos obteniendo los teoremas.

A: Hechos base = { *a(el,0,1), a(arquitecto,1,2), a(construyó,2,3), a(el,3,4), a(edificio,4,5)* }

Estos hechos base se introducen en el algoritmo como entrada siguiendo la relación 6–aria definida previamente y obtenemos:

Nivel 1:

ΔP := P = { p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,0,1,1),
p(n(X,arquitecto(X),gn(m,s),X:[humano],1,2,1),
p(v(X,Y,construir(X,Y),gn(_,s),(X:[humano],Y:[construcción]),2,3,1),
p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,3,4,1)
p(n(X,edificio(X),gn(m,s),X:[construcción],4,5,1) }

P:= ∅

Nivel 2:

ΔP_{pivot} *:= { p(sn(X,Z,existe(X,arquitecto(X), Z), gn(m,s),X:[humano],0,2,2),*
p(sn(X,Z,existe(X,edificio(X), Z), gn(m,s),X:[construcción],3,5,2) }

P := { p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,0,1,1),
p(n(X,arquitecto(X),gn(m,s),X:[humano],1,2,1),
p(v(X,Y,construir(X,Y),gn(_,s),(X:[humano],Y:[construcción]),2,3,1),
p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,3,4,1),
p(n(X,edificio(X),gn(m,s),X:[construcción],4,5,1) }

$\Delta P := \Delta P_{pivot}$

Nivel 3:

ΔP_{pivot} *:=*

{ p(sv(X,existe(Y,edificio(Y),construir(X,Y)),gn(m,s),(X:[humano],Y),2,5,3) }

P := { p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,0,1,1),
p(n(X,arquitecto(X),gn(m,s),X:[humano],1,2,1),
p(v(X,Y,construir(X,Y),gn(_,s),(X:[humano],Y:[construcción]),2,3,1),
p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,3,4,1),
p(n(X,edificio(X),gn(m,s),X:[vivienda],4,5,1),
p(sn(X,Z,existe(X,arquitecto(X),Z),gn(m,s),X:[humano],0,2,2),
p(sn(X,Z,existe(X,edificio(X), Z),gn(m,s),X:[construcción],3,5,2) }

$\Delta P := \Delta P_{pivot}$

Nivel 4:

$\Delta P_{pivot} :=$

{ p(s(existe(X,arquitecto(X),existe(Y,edificio(Y),construir(X,Y))),gn(_,_),(X,Y),0,5,4) }

P := { p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,0,1,1),
p(n(X,arquitecto(X),gn(m,s),X:[humano],1,2,1),
p(v(X,Y,construir(X,Y),gn(_,s),(X:[humano],Y:[construcción]),2,3,1),
p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,3,4,1),
p(n(X,edificio(X),gn(m,s),X:[vivienda],4,5,1),
p(sn(X,Z,existe(X,arquitecto(X),Z),gn(m,s),X:[humano],0,2,2),
p(sn(X,Z,existe(X,edificio(X), Z),gn(m,s),X:[construcción],3,5,2),
p(sv(X,existe(Y,edificio(Y),construir(X,Y)),gn(m,s),(X:[humano],Y),2,5,3) }

$\Delta P := \Delta P_{pivot} = \varnothing$

El conjunto final de teoremas derivado será:

P := { p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,0,1,1),
p(n(X,arquitecto(X),gn(m,s),X:[humano],1,2,1),
p(v(X,Y,construir(X,Y),gn(_,s),(X:[humano],Y:[construcción]),2,3,1),
p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,3,4,1),
p(n(X,edificio(X),gn(m,s),X:[vivienda],4,5,1),
p(sn(X,Z,existe(X,arquitecto(X),Z),gn(m,s),X:[humano],0,2,2),
p(sn(X,Z,existe(X,edificio(X), Z),gn(m,s),X:[construcción],3,5,2),
p(sv(X,existe(Y,edificio(Y),construir(X,Y)),gn(m,s),(X:[humano],Y),2,5,3)
p(s(existe(X,arquitecto(X),existe(Y,edificio(Y),construir(X,Y))),gn(_,_),(X,Y),0,5,4)
}

Con lo que queda demostrada la existencia de un hecho derivado de tipo *s* entre las posiciones inicial y final de la cadena.

En el ejemplo hemos podido constatar la aplicación de las restricciones gramaticales correspondientes:

a) cat=cat1+cat2, donde por unificación se obtiene la forma lógica de la categoría gramatical asociada.
b) est=est1+est2, que implica una concordancia en género y número entre los hechos.
c) ras=ras1+ras2, que implica una restricción semántica por concordancia según la ontología de rasgos semánticos.

6 Resolución de la extraposición

Este apartado está basado en: "Paralelismo sintáctico–semántico para el tratamiento de elementos extrapuestos en textos no restringidos" [Saiz97] presentado en la VIII Conferencia de la Asociación Española Para la Inteligencia Artificial celebrado en Torremolinos (Málaga) en noviembre, 1997.

6.1 El método

El método que presentamos en este trabajo intenta resolver la extraposición basándose en el paralelismo sintáctico–semántico para identificar el elemento extrapuesto. Consideramos que dos estructuras son paralelas si son isomórficas y tienen las mismas condiciones sintácticas y semánticas. En concreto, para el caso de la extraposición, diremos que el elemento extrapuesto y la traza son paralelos sintáctica y semánticamente si el elemento extrapuesto cumple las condiciones esperadas por la traza o el hueco dejado.

La idea principal que subyace en este nuevo método es el uso de la resolución semántica en la reconstrucción de la frase en lugar de usar las aproximaciones sintácticas. Así, la forma lógica del antecedente se identifica con el elemento extrapuesto. Cuando se producen problemas de ambigüedad, se utiliza un mecanismo de tipo semántico que pueda resolver un elevado número de casos.

El tratamiento del fenómeno lingüístico de la extraposición de elementos, necesita identificar en primer lugar cuáles son los constituyentes que han sido extrapuestos y cuál es su nueva posición, y en segundo lugar, cuál es la posición que ha quedado vacía en la oración de relativo (traza).

El paralelismo sintáctico–semántico nos ayudará a determinar las estructuras paralelas de forma automática (el elemento extrapuesto y la traza), mediante la aplicación incremental de una restricción de gramática Datalog con predicción top–down que completará la estructura perdida a través de un análisis de paralelismo inspirado en [Palomar96].

Durante el algoritmo *Semi–naive* comprobaremos si se ha derivado algún hecho pronombre relativo. Si esto ocurre, el sistema se encuentra alerta ante la presencia de una posible extraposición. En este mismo momento se pone en marcha un mecanismo de reconstrucción semántica de la oración de relativo que generará la forma lógica correspondiente.

Para simplificar la exposición del método usaremos una relación 3–aria como subconjunto de la 6–aria definida anteriormente. Esta relación contiene únicamente la categoría gramatical y la representación asercional del componente dentro de la cadena de entrada.

El método contiene las siguientes fases:

Fase 1: Identificar el pronombre relativo

> Identificar el pronombre relativo "*que*" y su posición en la oración. Para ello se consulta en la base de hechos la existencia de un hecho base *p(prel,M,M+1)*.

Fase 2: Definir los límites de la oración de relativo

> Detectar los límites de la oración de relativo. En una oración de relativo, los límites los marcan por una parte el pronombre relativo, y por otra la existencia de un elemento que ya no concuerda semánticamente con la oración de relativo sino que pertenece a la

oración principal. Dos elementos se dice que concuerdan semánticamente si, además de existir ambos hechos, ambos mantienen compatibilidad con el verbo, es decir, sus tipos semánticos son compatibles con las restricciones semánticas del verbo. Para conseguir este efecto nos basaremos en el método IRSAS (Incorporar Restricciones Semánticas en el Análisis Sintáctico) propuesto por [Moreno92a]. Una breve descripción de este método se expone en el Apéndice.

En el caso:

El jardinero plantó las flores que Juan regaló a María en el macetero del hall.

El inicio de la oración de relativo se marca por la existencia del pronombre relativo *"que"* y su fin se detecta al percibir que los rasgos semánticos del sintagma preposicional *"en el macetero del hall"* ya no concuerdan con los del verbo *regaló* sino con los de *plantó*.

Nótese que en este momento podría darse una ambigüedad semántica que no permitiera distinguir si el componente que hace de "frontera" pertenece o no a oración de relativo. Esto se debe definir correctamente en la ontología de rasgos semánticos para evitar errores. En cualquier caso, si se llega a una ambigüedad irresoluble se tomará el criterio de asociarlo a la oración de relativo (por proximidad) tal y como lo hace el propio entendimiento humano.

Desde este momento ya se sabe que debería encontrarse una oración completa entre los límites señalados.

Fase 3: Detectar la traza en la oración de relativo

Detectar dentro de los límites de la oración de relativo la ausencia de un sintagma (la *traza*). En castellano se puede producir una extraposición de dos formas:

a) Cuando el elemento “perdido” es el sujeto de la oración de relativo:

El soldado de plomo que (traza) *dispara al aire pertenece al ejército de infantería.*

b) Cuando el elemento perdido es un complemento del verbo en la oración de relativo:

El expediente que María perdió (traza) *está sobre su mesa.*

La detección del primer caso se produce porque se ha encontrado un hecho derivado del tipo *p(sv,N,P)* y no se ha encontrado ningún hecho derivado *p(sn,X,Y)*, con *X>=M+1* e *Y<=N* (es decir, entre el pronombre relativo y el sintagma verbal) que pudiera hacer las veces de sujeto de la oración. Esto puede ocurrir bien porque no se haya podido derivar ningún hecho sintagma nominal entre el hecho base pronombre relativo y el hecho derivado sintagma verbal o porque, aún habiéndose encontrado, no concuerde semánticamente con lo que debería ser el sujeto de ese sintagma verbal.

En este caso estamos buscando el sujeto siguiendo el orden lógico de la frase, es decir, siempre delante del verbo, sin entrar a resolver otro tipo de fenómenos lingüísticos que pudieran hacer variar este orden.

Para detectar la traza en el segundo caso, lo que se encontrará es un hecho base *p(v,N,N+1)* que por sus características sintácticas y de restricciones semánticas necesita la presencia de un sintagma nominal que actúe como complemento directo, y que sin embargo no se encuentra.

Tanto si estamos en el caso de una traza–sujeto o traza–complemento–del–verbo, el analizador tomará los rasgos semánticos del verbo para aplicárselos a la traza.

Fase 4: Detectar el elemento extrapuesto en su nueva posición

Buscar el elemento extrapuesto entre todos los hechos derivados que podrían serlo potencialmente, es decir hechos *p(sn,X,M)* que representan a todos los sintagmas

nominales situados inmediatamente a la izquierda del pronombre relativo representado por *p(prel,M,M+1)* y que sean paralelos semánticamente con el sintagma nominal que hemos definido como traza. Puesto que es posible que encontremos más de uno que cumpla con las restricciones sintácticas y semánticas hemos de aplicar un criterio para desambiguar. Para ello proponemos dos hipótesis de trabajo:

Hipótesis 1: Por similitud con el razonamiento humano desambiguaremos siempre tomando el hecho derivado que tenga la menor distancia de *X* a *M*. De esta forma, se ordenan todos los hechos derivados que concuerdan con el patrón definido anteriormente y de allí se extrae el menor.

Por ejemplo, en la frase:

0 El 1 tío 2 de 3 la 4 chica 5 que 6 trabaja 7 con 8 Juan 9 diseñó 10 el 11 edificio 12.

donde se ha detectado un hecho *p(prel,5,6)* y se sabe que el verbo *p(v,6,7)* necesita como sujeto un sintagma nominal con rasgo *[humano]*. Por lo tanto, las condiciones que debemos buscar en el candidato a elemento extrapuesto son: tener categoría gramatical de sintagma nominal, terminar en la posición 5 y tener rasgo *[humano]*. Tanto en el caso del *p(sn,4,5)* como en el *p(sn,0,5)* el rasgo que marca la cabecera de la estructura tienen la característica semántica *[humano]*, por lo que cualquiera de los dos sería candidato a elemento extrapuesto. Con esta primera hipótesis de trabajo decidimos que el candidato es el más corto, es decir, es *la chica* quien *trabaja con Juan* y no *el tío de la chica*

Hipótesis 2: Una forma más exacta de desambiguar (aunque de mayor complejidad) sería mediante la definición de una concordancia de rasgos semánticos gradual. De esta forma siempre tomaríamos el hecho derivado que concordando sintácticamente con un sintagma nominal y estando situado asercionalmente inmediatamente a la izquierda del pronombre relativo tenga semánticamente un grado de concordancia mayor.

Esto podría ocurrir en la frase:

$_0$El$_1$ hombre $_2$del $_3$ perro $_4$ lazarillo $_5$que $_6$pasea $_7$por $_8$el $_9$parque $_{10}$ ...

Aunque *pasear* puede aplicarse a un antecedente *[animal]* sería más exacto aplicarlo a un *[humano]*. Así podríamos determinar que el hecho derivado *p(sn,0,5)* es "más candidato" que el hecho derivado *p(sn,3,5)* puesto que éste sólo tiene la característica de *[animal]* y el primero es *[animal]* y es *[humano]*.

El problema que se plantea es que, en este caso, la fase de desambiguación entre los candidatos a elemento extrapuesto no debería realizarse hasta haber obtenido un hecho derivado *p(s,0,Z)* es decir, haber obtenido el hecho que nos indica que ya se ha analizado la frase completamente (se ha agotado el contexto de resolución de la extraposición) para asegurarnos de que ya no hay más candidatos. Sin embargo, con el primer método, sea cual sea el nivel del algoritmo Semi–naive actual, siempre que ya se hayan cumplido las fases 1 a 3 del algoritmo de reconstrucción semántica y se haya encontrado al menos un candidato que cumple con todas las restricciones, ya podremos hacer la reconstrucción semántica de la frase puesto que si posteriormente se derivan mas hechos candidatos éstos siempre tendrán una distancia *x* a *m* mayor. Sin embargo, si no se encuentra ninguno en este nivel habrá que intentarlo de nuevo en niveles posteriores.

Fase 5: Reconstrucción semántica de la oración.

Finalmente se reconstruye la forma lógica de la oración de relativo incluyendo toda la información semántica del elemento extrapuesto en la posición de la traza.

A continuación se muestran dos ejemplos completos de resolución semántica de elementos extrapuestos, cada uno de los cuales representa un caso en la clasificación de oraciones de relativo efectuada en el punto 2.2 :

6.2 Ejemplo de resolución de extraposición en oraciones de relativo de sujeto

0 El 1 perro 2 que 3 tenía 4 la 5 rabia 6 mordió 7 a 8 la 9 mujer 10

Entrada	*El perro que tenía la rabia mordió a la mujer*
Hechos Base	*a(el,0,1), a(perro,1,2), a(que,2,3), a(tenía,3,4), a(la,4,5), a(rabia,5,6), a(mordió,6,7), a(a,7,8), a(la,8,9), a(mujer,9,10)*
Derivación nivel 1	*p(det,0,1), p(n,1,2), p(prel,2,3), p(v,3,4), p(det,4,5), p(n,5,6), p(v,6,7), p(prep,7,8), p(det,8,9), p(n,9,10)*
Pronombre relativo identificado entre 2 y 3	
Derivación nivel 2	*p(sn,0,2), p(sn,4,6), p(sn,8,10)*
Derivación nivel 3	*p(sv,3,6), p(sp,7,10)*
Derivación nivel 4	*p(sv,6,10)*
Límites de la oración de relativo detectados entre 2 y 6	
Traza detectada entre 3 y 3	
Elemento extrapuesto detectado entre 0 y 2	
Reconstrucción semántica e inserción del hecho: *p(orel, 2, 6)*	
Derivación nivel 5	*p(sn,0,6)*
Derivación nivel 6	*p(s,0,10)*

En este ejemplo podemos ver cómo tras la derivación del nivel 1 en el algoritmo Semi–naive se identifica la existencia de un hecho derivado *pronombre relativo "que"* entre las posiciones 2 y 3. Esto ya pone en alerta al sistema indicándole que puede haber una extraposición. El próximo objetivo a conseguir es la identificación de los límites de *la oración*

de relativo. El límite inferior ya lo ha marcado el *pronombre relativo* pero hace falta identificar el límite superior. Tras el nivel de derivación 4 ya se encuentra que el hecho derivado *sintagma verbal* de la *oración de relativo* linda con otro hecho derivado *sintagma verbal*. Esto es señal de hemos encontrado los límites de la *oración de relativo* entre 2 y 6. En este momento se busca el hueco o *traza* en la *oración de relativo* a través de la semántica del *sintagma verbal* y se encuentra la falta de un *sujeto*. Por concordancia semántica se encuentra el elemento extrapuesto entre las posiciones 0 y 2. El hueco se rellena con el elemento extrapuesto y se reconstruye la forma lógica de la *oración de relativo* con su estructura semántica. La reconstrucción de la orel se efectuará igual que una oración simple con la salvedad de que a partir de ahora se comporta como un adjetivo.

A continuación veremos la reconstrucción semántica para este ejemplo usando la derivación de hechos representados por la relación P 6–aria:

Entrada:

A= {a(El,0,1), a(perro,1,2), a(que,2,3), a(tenía,3,4), a(la,4,5), a(rabia,5,6), a(mordió,6,7), a(a,7,8), a(la,8,9), a(mujer,9,10)}

Hechos derivados:

P1={

p(det(X,Y,Z,existe(X,Y,Z)),gn(m,s),_,0,1,1),
p(n(X,perro(X)),gn(m,s),[animal],1,2,1),
p(prel(X,Y,X&Y),gn(_,_),_,2,3,1),
p(v(X,Y,tener(X,Y)),gn(_,s),_,3,4,1),
p(det(X,Y,Z,existe(X,Y,Z)),gn(f,s),_,4,5,1),
p(n(X,rabia(X)),gn(f,s),[enfermedad],5,6,1),
p(v(X,Y,morder(X,Y)),gn(_,s),(X:[animal],Y:[animal]),6,7,1),
p(prep(X,Y),gn(_,_),_,7,8,1),
p(det(X,Y,Z,existe(X,Y,Z)),gn(f,s),_,8,9,1),
p(n(X,mujer(X)),gn(f,s),[humano],9,10,1) }

P2=P1 ∪ {

p(sn(X,Y,existe(X,perro(X)),gn(m,s),[animal],0,2,2),

p(sn(X,Y,existe(X,rabia(X)),gn(f,s),[enfermedad],4,6,2),

p(sn(X,Y,existe(X,mujer(X)),gn(f,s),[humano],8,10,2)}

P3=P2 ∪ {

p(sv(X,existe(Y,rabia(Y),tener(X,Y))),gn(_,s),(X:[entidad]),3,6,3),

p(sp(X,Y,existe(X,mujer(X),Y)),gn(f,s),[humano],7,10,3) }

P4=P3 ∪ {

p(sv(X,existe(Y,mujer(Y),morder(X,Y))),sin(_,s),X:[humano],6,10,4) }

En este momento se reconstruye semánticamente la oración de relativo y se añade al nivel 4 de derivación:

P4=P4 ∪ {

p(orel(existe(X,perro(X),existe(Y,rabia(Y),tener(X,Y)))),gn(_,_),_,2,6,4)}

P5=P4 ∪ {

p(sn(X,Z,existe(X,perro(X)&existe(Y,rabia(Y),tener(X,Y)),Z)),

gn(m,s),[animal],0,6,5)}

P6=P5 ∪ {

p(s(existe(X,perro(X)&existe(Y,rabia(Y),tener(X,Y)),

existe(Z,mujer(Z),morder(X,Z)))),gn(_,_),_,0,10,6) }

6.3 Ejemplo de resolución de extraposición en oraciones de relativo de complemento del verbo

$_0$ El $_1$ expediente $_2$ que $_3$ María $_4$ perdió $_5$ está $_6$ sobre $_7$ su $_8$ mesa $_9$

Entrada	*El expediente que María perdió está sobre su mesa*

Hechos Base	*a(el,0,1), a(expediente,1,2), a(que,2,3), a(María,3,4), a(perdió,4,5), a(está,5,6), a(sobre,6,7), a(su,7,8), a(mesa,8,9)*
Derivación nivel 1	*p(det,0,1), p(n,1,2), p(prel,2,3), p(n,3,4), p(v,4,5), p(v,5,6), p(prep,6,7), p(det,7,8), p(n,8,9)*
Pronombre relativo identificado entre 2 y 3	
Límites de la oración de relativo detectados entre 2 y 5	
Derivación nivel 2	*p(sn,0,2), p(sn,3,4), p(sn,7,9)*
Traza detectada entre 5 y 5	
Elemento extrapuesto detectado entre 0 y 2	
Reconstrucción semántica e inserción del hecho: *p(orel, 2, 5)*	
Derivación nivel 3	*p(sn,0,5), p(sp,6,9)*
Derivación nivel 4	*p(sv,5,9)*
Derivación nivel 5	*p(s,0,9)*

En este otro caso vemos cómo la detección de los límites de la *oración de relativo* se encuentran tras la derivación de nivel 1 al haber identificado ya la existencia de un hecho derivado *pronombre relativo "que"* entre las posiciones 2 y 3, y encontrar que el *verbo* de la *oración de relativo* linda con otro *verbo* comprobando semánticamente que ninguno funciona como auxiliar del otro. Esto nos indica que los límites de la *oración de relativo* están entre 2 y 5. Puesto que aún hay componentes por unificar en la *oración de relativo* tendremos que esperar hasta el nivel 2 de derivación para determinar el hueco o *traza* de la *oración de relativo*. Analizando la semántica del *verbo* se encuentra la falta de un complemento del verbo, el *objeto directo*. Por concordancia semántica se encuentra el elemento extrapuesto entre las posiciones 0 y 2. El hueco se rellena con el elemento extrapuesto y se reconstruye la forma lógica de la *oración de relativo* con su estructura semántica. En este caso nos saltamos un paso en el árbol de derivación puesto que hemos obtenido un hecho *oración de relativo*

partiendo del hecho derivado *sujeto*, el hecho derivado *verbo* y el hecho derivado *objeto directo*, sin haber derivado el hecho *sintagma verbal*. Esto lo hacemos así puesto que ya sabemos que no queda nada más por unificar dentro de la *oración de relativo* y nos ahorramos un paso.

7 Conclusiones

En este trabajo hemos presentado un estudio para el tratamiento de la extraposición a izquierdas de elementos en las oraciones de relativo utilizando programas Datalog y técnicas incrementales para su análisis. Para ello, en primer lugar, hemos realizado un estudio lingüístico del fenómeno de la extraposición contemplando los distintos casos que plantean tanto el idioma castellano como el inglés y confluyendo en una clasificación común que engloba los casos de extraposición en ambas lenguas. Por otra parte, hemos llevado a cabo un exhaustivo trabajo de recopilación de antecedentes para la resolución del problema mediante tratamiento computacional que fueron incluidos en el artículo [Ferrández97] presentado en el XIII Congreso de la Sociedad Española para el Procesamiento del Lenguaje Natural celebrado en Madrid en Julio 1997. Finalmente, hemos presentado un método de resolución de la extraposición a izquierdas basado en el paralelismo sintáctico–semántico de los componentes [Saiz97], trabajo que se presentó en la VIII Conferencia de la Asociación Española Para la Inteligencia Artificial celebrado en Torremolinos (Málaga) en noviembre de 1997.

8 Trabajos futuros

Partiendo del trabajo aquí expuesto pretendemos continuar nuestra línea de investigación con un tratamiento generalizado de la extraposición a larga distancia, basándonos para ello en el estudio de los fenómenos de dependencias ilimitadas [Pereira87]. Además, pretendemos tratar el fenómeno de la extraposición en textos no restringidos mediante la aplicación de análisis parciales basados en una gramática formal a través de un proceso de extracción de información relevante [Young97]. Estas técnicas de análisis parciales nos permitirán recuperar la información semántica de los textos no restringidos de forma eficiente sacrificando la completitud del análisis.

Referencias

[Allen95] Allen, J. *Natural Language Understanding*. 2nd ed. The Benjamin/Cummings Publishing Company, Inc. 1995

[Callejo97] Callejo, F. *Usos de que*. http://usuarios.bitmailer.com/fcallejo/que.htm.1997

[Casares94] Casares, J. *Diccionario ideológico de la lengua española*. Ed. Gustavo Gili, S.A. 1994

[Colmerauer78] Colmerauer, A. *Metamorphosis Grammars*. Lecture Notes in Computer Science, Springer–Verlag, pp 133–189, 1978

[Covington94] Covington, M. *Natural Language Processing for Prolog Programmers*. Prentice Hall. 1994

[Chomsky82] Chomsky, N. *Lectures on Government and Binding*, the Pisa Lectures, 2nd (revised) Edition. Foris Publications, Holland, 1982

[Chomsky88] Chomsky, N. *La nueva sintaxis: Teoría de Rección y Ligamiento*. 1988

[Dahl83] Dahl, V.; McCord, M. *Treating coordination in logic grammars*. American Journal of Computational Linguistics, 9, 69–91, 1983.

[Dahl84a] Dahl, V.; Abramson, H. *On gapping grammars*. Proc. Second International Conference on Logic Programming, Uppsala, Sweden, 1984

[Dahl84b] Dahl, V. *More on gapping grammars*. Proceedings International Conference on V Generation Computer Systems, Tokyo, 1984

[Dahl86] Dahl, V.; Saint–Dizier, P. *Constrained Discontinuous Grammars a linguistically motivated tool for processing language*. TR LCCR 86–11, Simon Fraser University, 1986

[Dahl88] Dahl, V. *Static Discontinuity Grammars for Government–Binding Theory*. LCCR TR 88–22, Simon Fraser University, 1988

[Dahl89] Dahl, V. *Discontinuous Grammars*. Computational Intelligence, 5(4): 161–179, 1989

[Dahl90] Dahl, V.; Popowich, F.; *Parsing and Generation with Static Discontinuity Grammars*. New Generation Computing, 8. 1990

[Dahl94a] Dahl, V. *Natural Language Processing and Logic Programming*. The Journal of Logic Programming, vol. 19 nº20, 1994

[Dahl94b] Dahl, V.; Tarau P.; Huang Y; *Datalog Grammars*. Proc. Of the GULP–PRODE'94. V. 2. ed. UPV, 1994

[Dahl95] Dahl, V.; Tarau P.; Moreno, L.; Palomar, M.; *Treating coordination with Datalog Grammars*. Computational Logic for natural Language Processing. (CLNLP–95). Edinburg. Scotland. 1995

[Ferrández95] Ferrández, A.; Moreno, L.; Palomar, M. *Un formalismo para el tratamiento gramatical de la coordinación: Gramática de Unificación de Huecos.* Novatica, 115. 1995

[Ferrández97] Ferrández, A.; Peral, J.; Martínez–Barco, P.; Sáiz, M.; Romero, R. *Resolución de la extraposición a izquierdas con las Gramáticas de unificación de Huecos.* Procesamiento del Lenguaje Natural, revista nº 21. 1997

[Gazdar89] Gazdar, G.; Mellish, C. *Natural Language Processing in Prolog: An Introduction to Computational Linguistics*. Addison–Wesley Publishing Company. 1989

[Groenink95] Groenink, A.. *Literal Movement Grammars*. 1995

[Huang94] Huang, Y–N.; Dahl, V.; Han, J. *Fact Updates in Logic Databases*. Report interno no publicado. School of Computing Science, Simon Fraser university, Canada. 1994

[Kroch86] Kroch, A.S.; Joshi, A.K. *Analysing Extraposition in a TAG*. Ojeda Huck, editor, Syntax and Semantics: *Discontinuous Constituents*. Acad Press, New York. 1986

[Lázaro85] Lázaro, F.; Tusón V. *Curso de Lengua Española*. Ed. Anaya. 1985

[McCord91] McCord, M. *Slot Grammar*. IBM Thomas J. Watson Research Center. 1991

[Moreno92a] Moreno, L.; Andrés, F.; Palomar, M. *Incorporar Restricciones Semánticas en el Análisis Sintáctico: IRSAS*. Procesamiento del Lenguaje Natural n.12. 1992

[Moreno92b] Moreno, L.; Palomar, M. *Semantic Constraints in a Syntactic Parser: Queries–Answering to Database*. Database and Expert Systems Applications. Springer–Verlag. 1992

[Moreno93] Moreno, L.; *Formalismos lógicos para el análisis e interpretación oracional del lenguaje natural*. Tesis doctoral. (FI– Universidad Politécnica de Valencia). 1993

[Moreno96] Moreno, L.; Molina A. L*a ambigüedad en los distintos niveles de análisis del Procesamiento del Lenguaje Natural*. Boletín de AEPIA, nº6, 71–83, 1996

[Moreno97] Moreno, L.; Palomar, M.; Molina A.; *Gramáticas Datalog Extendidas: una nueva aproximación*. APPIA–GULP–PRODE'97. Join Conference on Declarative Programming. Grado. Italia. Junio 1997

[Palomar93] Palomar, M.; Moreno, L.; López, V. *Aproximación Metagramatical Sintáctico Semántica de la Coordinación*. Procesamiento del Lenguaje Natural, nº13, pag. 135–144. 1993

[Palomar95] Palomar, M.; Ferrández, A.; Moreno, L. *Aportaciones a la Resolución de la elipsis en la coordinación*. Procesamiento del Lenguaje Natural, nº16, 1995

[Palomar96] Palomar, M. *Aportaciones a la Resolución de la elipsis en lenguaje natural utilizando técnicas incrementales*. Tesis Doctoral (FI– Universidad Politécnica de Valencia). 1996

[Pereira80] Pereira, F.; Warren, D. *Definite Clause Grammars for Language Analysis– A Survey of the Formalism and a Comparison with Augmented Transition Networks*. Artificial Intelligence, vol. 13. 1980

[Pereira81] Pereira, F.C.N. *Extraposition Grammars*. Computational Linguistics, 7(4), 1981

[Pereira87] Pereira, F.; Shieber, S. *Further Topics in Natural–Language Analysis. Prolog and Natural–Language Analysis*. Lecture notes nº10. CLSI. 1987

[Pollard84] Pollard, C.J. *Generalizad Phrase Structure Grammars, Head Grammars, and Natural Language*. Ph. D. thesis, Standford University. 1984

[Popowich86] Popowich, F.P. *Unrestricted gapping grammars*. Computational Intelligence Journal, vol 2, pp 28–53, 1986

[Popowich89] Popowich, F.P. *Tree Unification Grammar*. In Proc. of 27th Anual Meeting of A. for Computational Linguistics. Vancouver, Canada, 1989

[Popowich93] Popowich, F.P. *Lexical Characterization of Local Dependencies with Tree Unification Grammar*. Technical Report CSS–IS TR 93–13, SFU, Burnaby, B.C. Canada, 1993

[Saiz97] Saiz, M.; Martínez–Barco, P.; Palomar, M. *Paralelismo sintáctico–semántico para el tratamiento de elementos extrapuestos en textos no restringidos*. CAEPIA, 1997

[Sanchez95] Sánchez, F. *Desarrollo de un etiquetador morfosintáctico para el español*. Procesamiento del Lenguaje Natural, nº17. 1995

[Sag97] Sag, I.A. *English Relative Clause Constructions*. Journal of Linguistics. 1997

[Ullman89] Ullman, D.J. *Principles of Database and Knowledge–Base Systems*. Computer Science Press. Vol I y II. 1989

[Young97] Young, S.; Bloothooft, G. *Corpus–Based Methods in Language and Speech Processing*. Kluwer Academic Publishers. 1997

Apéndice. Método IRSAS.

El método IRSAS [Moreno92a] [Moreno92b] [Moreno93] se basa en la desambiguación de palabras teniendo en cuenta el contexto semántico de la oración. Para ello cada objeto del universo se encuentra clasificado por sus propiedades en un grupo semántico. Así cada objeto está formado por *tipo* y *referente*. El *tipo* es un conjunto de rasgos que clasifican al objeto y el *referente* que identifica al objeto genérico o individual. Por ejemplo:

El tipo de *perro* está formado por una serie de rasgos semánticos: *animal, mamífero, macho, etc.* y su referente es *perro.* El tipo de *comía* está definido por *acción, etc.* y su referente es *comer*

La clasificación de la jerarquía de rasgos del universo (*ontología de los rasgos semánticos*) se define mediante la aplicación de dos relaciones básicas:

a) Las relaciones de división y herencia: por las que un rasgo se puede dividir en subrasgos que heredan del tipo padre sus rasgos semánticos, aunque añadiendo nuevas características: un *ser vivo* puede ser *animal* o *vegetal.* Un *animal* puede ser *macho* o *hembra*, etc... Esta relación se denota como:

   $rasgo \Leftrightarrow rasgo1$ v $rasgo2$ v ...$rasgoN$

   donde v es una disyunción exclusiva.

b) Las relaciones de implicación: por las que un rasgo específico puede implicar a otro rasgo más genérico. Así se pueden definir, por ejemplo, relaciones del tipo: un *perro* es un *cazador*. Se denota como:

   $rasgoA \Rightarrow rasgoB$

Una vez definida correctamente la ontología de rasgos siguiendo las relaciones anteriores, el método IRSAS las almacena en tres grafos:

a) El grafo de herencias, donde se relaciona cada rasgo con sus herederos.
b) El grafo de incompatibilidades, donde quedan relacionados aquellos rasgos que por formar parte de la disyunción exclusiva de una herencia no podrían darse simultáneamente en un mismo tipo semántico (ningún tipo semántico podría contener a la vez los rasgos *animal* y *vegetal*).
c) El grafo de implicaciones, que mantiene las relaciones de implicación definidas.

Finalmente, partiendo de estos grafos indicados y mediante la aplicación de una serie de condiciones de consistencia entre listas de rasgos el método IRSAS es capaz de determinar la compatibilidad semántica entre dos términos.

La ontología de rasgos con la que estamos trabajando es una versión adaptada de la que se usa en [Palomar96] que es a su vez un subconjunto de la definida en el Plan general de clasificación ideológica del diccionario de J. Casares [Casares94]:

entidad ⇔ concreto v abstracto
entidad ⇔ individual v colectiva
abstracto ⇔ temporal v espacial v mental
mental ⇔ estado v acción
concreto ⇔ orgánico v inorgánico
orgánico ⇔ vegetal v animal
animal ⇔ humano v no_humano
animal ⇔ macho v hembra
inorgánico ⇔ geografía v física
geografía ⇔ ciudad v montaña
física ⇔ sólido v líquido v gas
líquido ⇔ bebible v no_bebible
sólido ⇔ comestible v no_comestible
sólido ⇔ construcción v no_construcción

estado ⇔ salud v enfermedad

animal ⇒ sólido

ciudad ⇒ sólido

La implementación original del método se puede ver en [Moreno92a] [Moreno92b] [Moreno93].

Patricio M. Martínez Barco VºBº Manuel Palomar Sanz