INTERPRETACIÓN DE LA COMPARACIÓN EN CONSULTAS A UNA BASE DE DATOS GEOGRÁFICA A TRAVÉS DE LA LÓGICA

Moreno, L.;Palomar, M.;Pastor, M.

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia

RESUMEN

El presente estudio se encuentra desarrollado dentro de un sistema capaz de responder consultas en lenguaje natural a una Base de Datos. El sistema procesa consultas en lenguaje castellano que serán evaluadas en una Base de Datos geográfica. El sistema se encuentra implementado en Prolog, concretamente en BIM-PROLOG, y puesto en marcha en un ordenador SUN 3/80. En este trabajo se aborda el análisis pragmático de los comparativos dentro de este sistema

1. INTRODUCCIÓN

El sistema somete las oraciones a un proceso compuesto por una serie de fases. Estas fases de procesamiento de la oración hasta poder ser evaluada y conseguir una respuesta son: análisis léxico, análisis sintáctico, análisis semántico y análisis pragmático. Este trabajo se centra en el Análisis Pragmático de los comparativos, las otras fases de procesamiento de estos pueden verse en [MORENO, 92b].

A partir del conjunto de tokens generados por el analizador léxico conseguiremos el árbol sintáctico asociado a la oración. Esta operación la lleva a término el Analizador Sintáctico, el cual tiene como objetivo comprobar que la oración puede ser generada a partir de las reglas gramaticales definidas mediante una Gramática Lógico Modular de M. McCord ([MCCORD, 85], [VACAN, 86] y [WALKER, 89]) si es así, obtendremos como salida el árbol sintáctico.

El Analizador Semántico, a partir de la información contenida en el árbol sintáctico, construirá la Forma Lógica asociada a la oración, tras combinar, por medio de operadores lógicos, las formas lógicas parciales de los constituyentes de dicha oración.

El Análisis Pragmático es el proceso por el cual la Forma Lógica asociada a la oración (resultante del análisis semántico), es transformada en su equivalente Forma Lógica en términos de la Base de Datos. Esta última será directamente evaluada sobre la Base de Datos, proporcionando la respuesta asociada a la consulta realizada por el usuario.

2. ESTUDIO DE LAS FORMAS LÓGICAS.

Los comparativos incrementan cuantitativamente el lexema al que modifican. Esta 'gradación' es propia tanto de los adjetivos, como de los sustantivos.

En el **caso del adjetivo**, tanto en su función de modificador del sintagma nominal como de atributo, puede incrementarse con un elemento de referencia cuantitativa, que será un adverbio de intensidad (mas, menos, tan), aunque a veces el lexema del adjetivo presenta un significante particular para expresar su combinación con el contenido del adverbio, como por ejemplo: mayor (más grande), menor (más pequenyo), mejor (más bueno), peor (más malo), etc.

En el **caso del sustantivo**, este puede incrementarse con el adverbio de intensidad (más, menos), o con adjetivos indefinidos (tanto y sus variaciones de género y número).

Sintácticamente los comparativos realizan la misma función que cualquier otro adjetivo. Generalmente, cuando aparece un comparativo en la oración, aparece también un segundo elemento que es la **coordinación comparativa**. La coordinación se lleva a cabo mediante las conjunciones **que, como** y la preposición **a**: mas grande que, tan grande como, superior a...Los dos objetos que se comparan deben tener el mismo tipo semántico asociado (sobre el método desarrollado sobre tipos semánticos ver [MORENO, 92a]).

En las Formas Lógicas correspondientes a los comparativos podemos observar características comunes entre éstas atendiendo a:

- los elementos que se comparan
- la Base de la comparación.

Siempre que se comparan dos elementos estos son o bien dos entidades, o dos atributos de entidades, o una entidad con el «resto», o un atributo con el «resto», o bien un resultado numérico con una constante numérica. En las frases en las cuales se hace una comparación con el «resto», hay que tener en cuenta que entidades o atributos forman el «resto» de la comparación. La Base de la comparación es un adjetivo calificativo o una propiedad que tienen los elementos que se comparan. Teniendo en cuenta estos parámetros, elementos comparados y base de comparación, diferenciaremos las Formas Lógicas de los comparativos, realizando la siguiente clasificación:

(a). DOS ENTIDADES:

a.1. Si comparamos dos entidades y la base de comparación es un adjetivo calificativo, la Forma Lógica es la siguiente:

comp(G,B,X,Y)

donde G es el grado de comparación, B es un adjetivo, y X e Y son dos entidades.

Δ Que rios son mas grandes que el rio Turia ?
preg(Y,rio(Y)&ex(rio(X)&X=turia,**comp**(mas,grande,Y,X)))

a.2. Si comparamos dos entidades y la base de comparación es una propiedad, la Forma Lógica es la siguiente:

comp(G,num(Z,P,N),X,Y)

donde G es el grado de comparación, la Base de comparación es una propiedad que puede medirse numericamente y X e Y serán dos entidades o atributos de entidades.

Δ Que rios tienen mas caudal que el rio Turia ?
preg(X,rio(Y)&ex(rio(X)&X=turia,

comp(mas,num(U,caudal(U)&tener1(Y,U),Z),Y,X)))

(b). DOS ATRIBUTOS DE ENTIDADES:

b.1. Si comparamos dos atributos de entidades y la base de comparación es un adjetivo calificativo , la Forma Lógica es la siguiente:

comp(G,B,X,Y)

donde G es un adverbio de intensidad, B es un adjetivo y X e Y son atributos de entidades.

Δ Dime los rios de longitud mas grande que la longitud que tiene el rio Turia.
preg(X, ex(km(Z) & ex(rio(U) & U=turia, tener1(U,Z)), ex(km(Y) &
comp(mas,grande,Y,Z), rio(X) & de(Y,X))))

(c). RESULTADO NUMERICO CON UNA CONSTANTE NUMERICA:

c.1. Si comparamos un resultado numérico con una constante numérica y la base de comparación es un adjetivo calificativo , la Forma Lógica es la siguiente:

comp(G,B,X,Num)

donde B es un adjetivo, X un atributo de una entidad y Num representa a la constante numérica que se compara con el atributo X.

Δ Dime los rios de longitud mayor que 200 kilometros.
preg(X,num(Z,km(Z)&ex(km(Y)&**comp**(mas,grande,Y,Z),rio(X)&de(Y,X)),200))

c.2. Si comparamos un resultado numérico con una constante numérica y la base de comparación es una propiedad , la Forma Lógica es la siguiente:

comp(G,num(X,P,N),N,Num)

donde G es un adverbio de intensidad y Num es la constante numérica que se compara con otra constante numérica N resultante de evaluar la base de comparación num(X,P,N). Siendo X una variable representando una atributo o una entidad cuyo rango y restricciones se especifican en la Forma Lógica representada por P.

Δ Que rios recorren mas de 2 autonomias ?
preg(Z,rio(Z)&**comp**(mas,num(Y,aut(Y)&pasar(Z,Y),X),X,2))

(d). ENTIDAD CON EL RESTO:

d.1. Si comparamos una entidad con el resto de entidades y la base de comparación es un adjetivo calificativo , la Forma Lógica es la siguiente:

comp(G,B,X,resto)

donde G es un adverbio de intensidad, B es un adjetivo y X es una entidad que se compara con el resto de entidades del mismo tipo semántico. Luego «resto» está representando al colectivo de entidades con idénticas restricciones a las que esté sometida la entidad objeto de la comparación.

Δ Que rio es mas grande ?

preg(X,rio(X)&**comp**(mas,grande,X,resto))

d.2. Si comparamos una entidad con el resto de entidades y la base de comparación es una propiedad , la Forma Lógica es la siguiente:

comp(G,num(Z,P,N),X,resto)

donde G es un adverbio de intensidad, la base de comparación es una propiedad que puede medirse cuantitativamente y X es una entidad que se compara con el «resto».

Δ Que rio recorre mas autonomias ?

preg(X,rio(X)&**comp**(mas,num(Z,aut(Z)&pasar(X,Z),Y),X,resto))

(e). ATRIBUTO CON EL RESTO:

e.1. Si comparamos el valor de un atributo de una entidad con los valores de ese mismo atributo para el resto de entidades y la base de comparación es un adjetivo calificativo , la Forma Lógica es la siguiente:

comp(G,B,X,resto)

donde G es un adverbio de intensidad, B es un adjetivo y X es un atributo.

Δ Dime el rio de longitud mayor.

preg(X,ex(km(Y)&**comp**(mas,grande,Y,resto),rio(X)&de(Y,X)))

3.TRANSFORMACIÓN Y EVALUACIÓN DE LAS FORMAS LÓGICAS

3.1 LLAMADA GENERAL AL MÓDULO PRAGMÁTICO

Después de haber realizado el Análisis Sintáctico_Semántico de una frase obtenemos la Forma Lógica correspondiente. Dicha Forma Lógica será la entrada al Módulo Pragmático, obteniendo como resultado:

a. Salidas intermedias: Forma Lógica simplificada y Forma Lógica en términos de la Base de Datos.

b. Salida final: La respuesta a la pregunta introducida por el usuario.

El predicado principal del Módulo Pragmático es el siguiente:

```
procesa4(FL):-
    mark(X),
```

```
preprocesa(FL,FL1,Cxto,Amb),
cut(X),
qtrans(FL1,Cxto,Amb,Q),
write('Forma logica en terminos de la Base de Datos...'),
write(Q),
write('Respuesta...'),
call(Q).
```

Primero se realiza un **preproceso** de la Forma Lógica que consta de las siguientes partes:

- Eliminar cualquier aparición de los predicados «ser(X,Y)», «nombre(X,Y)» y «ser_pasiva(X,Y)» de la Forma Lógica.

- Construir la Lista de Variables Tipificadas (o Contexto). Esta será una lista formada por elementos de la forma:

  Variable:tipo,

  es decir, una «Variable» que aparezca en la Forma Lógica, el operador «:» y el «tipo» de la variable. La Lista de Variables Tipificadas estará formada por tantos elementos como variables distintas aparezcan en la Forma Lógica, y el tipo le será asignado dependiendo del functor del cual sea argumento la correspondiente variable.

  Consultando esta lista podemos conocer en cualquier momento el tipo de una variable determinada para poder ver si es una variable asociada a una entidad o a un atributo. El poder conocer el tipo de cualquier variable también nos permite determinar la traducción correcta de un predicado cuyos argumentos, según la consulta, pueden ser distintos tipos de elementos. (Ej. «desembocar(X,Y)» siendo X de tipo río, tendrá una traducción distinta dependiendo de si Y es de tipo mar o de tipo autonomía).

- Proceso de Simplificación de la Forma Lógica que nos permite reducirla, y por tanto reducir el tiempo de evaluación de la Forma Lógica en términos de la Base de Datos. Aprovechando este proceso detectamos si en la frase hay alguna «comparación con el resto» (es decir, se compara un atributo o entidad con el resto de atributos o entidades).

- Si se ha detectado alguna «comparación con el resto» se procede a buscar el «Ambito» donde se evaluará el resto, es decir, se construye la expresión que debe cumplir cualquier elemento que forme parte del «resto».

  A continuación llamamos al predicado «**qtrans**» el cual se encarga de transformar la Forma Lógica, resultante del preproceso, a la Forma Lógica en términos de la Base de Datos. Y por último se procede a evaluar la Forma Lógica en términos de la Base de Datos, mediante **call(Q)**, obteniendo la respuesta.

3.2 TRANSFORMACIÓN DE PREDICADOS.

El predicado «qtrans» se encargará de transformar la Forma Lógica simplificada a otra en términos de la Base de Datos. Dicho predicado tendrá cuatro argumentos: los tres primeros representan las entradas y el último representa el resultado:

- 1º Forma Lógica simplificada.

- 2º Lista de Variables tipificadas (obtenida en el preproceso).

- 3º Ambito en el cual hay que evaluar el resto en caso de existir una comparación con el resto.

- 4º Forma Lógica en términos de la Base de Datos dispuesta a ser evaluada.

veamos el codigo Prolog para dicho predicado:

```
qtrans(P,Cxto,Amb,P):-
    (var(P);
    atomic(P)).
qtrans(P,Cxto,Amb,Q):-
    qt(P,Cxto,Amb,Q),
    !.
qtrans(P,Cxto,Amb,Q):-
    P=..[Pred|Args],
    (Pred = sino;Pred = preg;Pred = &;Pred = cant;Pred = '='),
    qtranslist(Args,Cxto,Amb,Args1),
    Q=..[Pred|Args1].
```

donde para el predicado 'qtranslist' tenemos el siguiente código:

```
qtranslist([P|L],Cxto,Amb,[Q|M]):-
   qtrans(P,Cxto,Amb,Q),
   qtranslist(L,Cxto,Amb,M).
   qtranslist([],Cxto,Amb,[]).
```

predicado «qt» es el que transforma un predicado de la Forma Lógica a otro predicado en términos de la Base de Datos. Es el predicado mas íntimamente ligado a la Base de Datos sobre la cual se realiza el Análisis Pragmático. Veamos un ejemplo:

```
qt(desembocar1(X,Y),Cxto,_rio(X,_,_,_,_,_,Y)):- tipo_var(Y,mar,Cxto), !.
```

Se podría interpretar de la siguiente manera: El predicado «desembocar1(X,Y)» donde X es de tipo río e Y es de tipo mar, lo transformamos por la tupla «rio(X,_,_,_,_,_,Y)» de la Base de Datos. Análogamente tendremos un predicado «qt» por cada forma de cita que tengamos almacenada en el Diccionario.

3.3 TRANSFORMACIÓN DE LAS COMPARATIVAS ENTRE 2 ELEMENTOS.

En la clasificación de las formas lógicas asociadas a las comparaciones dada en el punto 2.3, aparecen cinco grupos. Los dos últimos grupos hacen referencia a las comparaciones con el 'resto' que requieren un tratamiento bien diferenciado al de los otros tres primeros grupos, por lo que dejaremos para el punto siguiente las comparaciones con el 'resto'.

La Forma Lógica asociada a la comparación, «comp(G,B,E1,E2)», se transformará según los casos en un término que entrará a formar parte de la Forma Lógica en términos de la Base de Datos.

Este término normalmente estará formado por predicados de la Base de Datos y el predicado «grado(G,E1,E2)» (donde G es el grado de comparación, y E1 y E2 son los elementos que se compararán).

Los predicados asociados al Grado de la comparación son:

```
cambia_grado(mas,pequenyo,menos).
cambia_grado(mas,corto,menos).
cambia_grado(menos,pequenyo,mas).
cambia_grado(menos,corto,mas).
cambia_grado(X,_,X).
```

Cuando la Forma Lógica sea evaluada, el predicado «grado» se interpretará de acuerdo a las siguientes reglas:

```
grado(mas,X,Y):-
    X>Y.
grado(menos,X,Y):-
    X<Y.
grado(tan,X,Y):-
    X=Y.
```

Antes de pasar a ver el código de las reglas Prolog que traducirán las Formas Lógicas asociadas a las comparaciones, pasaremos a comentar los **predicados no predefinidos** en Prolog que en ellas aparecen:

- tipo_var(X,Tipo,Cxto): se hará cierto siempre que X tenga asociado el «Tipo» indicado. Si X es una variable habrá que comprobar que el elemento «X:Tipo» pertenece a la Lista de Variables Tipificadas «Cxto». Si X es un átomo, su tipo se comprobará con la ayuda de las tuplas de la Base de Datos.

- busca_tipo(X,Tipo,Cxto): devuelve el «Tipo» asociado a X.

- es_entidad(X,Cxto): el predicado se hará cierto si X tiene asociado el tipo de una entidad (rio, ciudad, mar, aut o sist_m).

- es_atributo(X,Cxto): el predicado se hará cierto si X tiene asociado el tipo de un atributo (km, km2, habit, caudal).

- elemento(X,Lista): comprueba si X pertenece a la «Lista».

- cambiar(P,[Ele1:Ele1',Ele2:Ele2',...],Q): devuelve en Q el predicado P, habiendo sustituido previamente cualquier ocurrencia de los elementos Ele1', Ele2', ... por los elementos Ele1, Ele2, ... respectivamente. Este predicado es muy útil cuando queremos construir un predicado idéntico a otro pero con distintas variables. Cuando se compara una propiedad entre dos elementos:

```
comp(G,Propiedad,X,Y),
```

dicha «Propiedad» siempre viene expresada en términos del primer elemento, X, nosotros nos serviremos del predicado «cambiar» para formar una expresión igual pero en términos del elemento Y, pudiendo comparar así las propiedades de los dos elementos.

• num_cardinal(Z,P,N): Obtiene en la variable N el número de entidades representadas por la variable Z que cumplan la expresión P.

El código que, atendiendo a la clasificación del apartado 2.3, trata este tipo de comparativas es:

(a). DOS ENTIDADES:

• Para el subgrupo a.1, cuya Forma Lógica es comp(G,B,X,Y) donde B es un adjetivo, tenemos el siguiente código cuando las entidades a comparar son dos ríos:

```
qt(comp(G,B,X,Y),Cxto,_,FL):-
  tipo_var(X,rio,Cxto),
  tipo_var(Y,rio,Cxto),
  cambia_grado(G,B,G1),
  ((elemento(B,[grande,largo,pequenyo,corto])) ->
    FL=(rio(X,_,L1,_,_,_,_) & rio(Y,_,L2,_,_,_,_) & grado(G1,L1,L2)) ;
    ((B=caudaloso) ->
      FL=(rio(X,C1,_,_,_,_,_) & rio(Y,C2,_,_,_,_,_) & grado(G1,C1,C2)))), !.
```

Se realiza la comprobación del tipo de los elementos que se van a comparar, y se normaliza el Grado con respecto a la Base de comparación. A continuación tenemos que saber que cualidad de las entidades hay que comparar, en este caso viene definida por un adjetivo que cualifica a un atributo de las entidades. La Forma Lógica Traducida (FLT) resultante constará de las dos tuplas correspondientes a las entidades que se comparan, con el objeto de extraer los valores de los atributos determinados por la cualidad que se compara, y el predicado «grado» que, dependiendo del grado de comparación normalizado, realizará la operación de comparación entre los dos valores numéricos.

• Para el subgrupo a.2, cuya Forma Lógica es comp(G,num(Z,P,N),X,Y) si Z es una entidad, tenemos el siguiente código:

```
qt(comp(G,num(Z,P,N),X,Y),Cxto,_,FL):-
  es_entidad(X,Cxto),
  es_entidad(Z,Cxto),
  cambiar(P,[Zeta:Z,Y:X],Q),
  busca_tipo(X,Tipoe,Cxto),busca_tipo(Z,Tipoa,Cxto),
  append(Cxto,[Y:Tipoe,Zeta:Tipoa],Cxto2),
  !, qtrans(P,Cxto,_,P1), !, qtrans(Q,Cxto2,_,Q1),
  FL1=(num_cardinal(Z,P1,N) & num_cardinal(Zeta,Q1,N1) & grado(G,N,N1)),
  ((var(X)) ->
```

```
(((Tipoe=aut) -> Ant=com_aut(X,_,_) ;
 ((Tipoe=mar) -> Ant=mar(X) ;
  ((Tipoe=rio) -> Ant=rio(X,_,_,_,_,_,_) ;
   ((Tipoe=sist_m) -> Ant=sist_m(X) ;
    ((Tipoe=ciudad) -> Ant=ciudad(X,_,_))))),
 FL=(Ant & FL1)) ;
FL=FL1), !.
```

Aquí la Base de la Comparación no es un adjetivo, sino que se trata de un término asociado a la propiedad de las entidades que se van a comparar. Las propiedades vienen plasmadas como «num(Z,P,N)» donde «N» es el número de «Z's» que cumplen «P». Por ejemplo, si la propiedad a comparar es «pasar por comunidades» que tienen dos ríos entonces N es el número de comunidades, y Z's son las comunidades que cumplen P, es decir, por las que pasa un río. La Forma Lógica traducida se compondrá de dos predicados «num_cardinal», uno para cada entidad, y el predicado «grado». Los predicados «num_cardinal», tendrán la misión, en el momento de la evaluación, de obtener el número de entidades que cumplen «P_traducido» para cada entidad que se compara; ambos números son los que intervendrán posteriormente en la comparación efectuada por «grado». Pero la propiedad (num(Z,P,N)) a comparar viene expresada en función del primer elemento que se compara (X), por lo que tenemos que construir una propiedad idéntica pero en función del segundo elemento que se compara (Y).

Si Z es atributo entonces el código para el predicado «qt» será el siguiente:

```
qt(comp(G,num(Z,P,N),X,Y),Cxto,_,FL):-
    es_atributo(Z,Cxto),
    es_entidad(X,Cxto),
    es_entidad(Y,Cxto),
    cambiar(P,[Zeta:Z,Y:X],Q),
    busca_tipo(X,Tipoe,Cxto),
    busca_tipo(Z,Tipoa,Cxto),
    append(Cxto,[Y:Tipoe,Zeta:Tipoa],Cxto2),
    !, qtrans(P,Cxto,_,P1), !, qtrans(Q,Cxto2,_,Q1),
    FL=(P1 & (N=Z) & Q1 & (N1=Zeta) & grado(G,N,N1)),!.
```

También nos podemos encontrar en el caso en que Z sea un atributo. La Base de la Comparación serán propiedades como «tener caudal». Esto implica que el predicado «grado» en el momento de la evaluación comparará dos valores de atributos de entidades. En este caso «N», en «num(Z,P,N)», será simplemente el valor del atributo representado por la variable «Z» en el predicado «P». Tras la preparación del predicado P, para cada una de las entidades «X» e «Y», y sus respectivas transformaciones, se construye la Forma Lógica traducida. Esta incorpora los predicados «P-traducidos» de los cuales se extraerán los valores de los atributos que comparará el predicado «grado».

(b). DOS ATRIBUTOS DE ENTIDADES:

- Para este grupo la Forma Lógica será comp(G,B,X,Y) donde G es un adverbio de intensidad, B es un adjetivo, y X e Y son atributos. El código correspondiente a la transformación será de la forma:

```
qt(comp(G,B,X,Y),Cxto,_,FL):-
  es_atributo(X,Cxto),
  busca_tipo(X,Tipoa,Cxto),
  cambia_grado(G,B,G1),
  ((Tipoa=caudal) -> Ant=rio(_,X,_,_,_,_) ;
   ((Tipoa=km) -> Ant=rio(_,_,X,_,_,_) ;
    ((Tipoa=km2) -> Ant=com_aut(_,X,_) ;
     ((Tipoa=habit) -> Ant=ciudad(_,X,_))))),
  FL=(Ant & grado(G1,X,Y)), !.
```

En este caso «X» e «Y» son dos atributos, por lo tanto la Forma Lógica Traducida constará de una tupla de la B.D. con el objeto de instanciar la variable X a un valor del atributo correspondiente, seguido del predicado grado que realizará la comparación en el momento de la evaluación. El código anterior también contempla el caso en que la comparación sea entre un atributo y una constante numérica.

(c). RESULTADO NUMÉRICO CON UNA CONSTANTE NUMÉRICA:

- Las Formas Lógicas correspondientes al subgrupo c.1 se tratan con el mismo código desarrollado para tratar el grupo de comparativos anterior, es decir el grupo (b).

- La Forma Lógica es comp(G,num(X,P,N),N,Num) donde X es atributo, perteneciente al subgrupo c.2, se transformará a través del siguiente código:

```
qt(comp(G,num(X,P,N),N,Num),Cxto,_,FL):-
      es_atributo(X,Cxto),
  !, qtrans(P,Cxto,_,Q),
  FL=(Q & (N=X) & grado(G,N,Num)), !.
```

Los elementos de la comparación son «N» y «Num». En este caso «X» es un atributo, por lo tanto, «N» será el valor correspondiente a ese atributo que cumple «P». De esta forma, «grado» se encargará de comparar «N» con la constante numérica «Num» en el momento de la evaluación.

Si en la Forma Lógica X es entidad, el código para la transformación será:

```
qt(comp(G,num(X,P,N),N,Num),Cxto,_,FL):-
  es_entidad(X,Cxto),
  !, qtrans(P,Cxto,_,Q),
  FL=(num_cardinal(X,Q,N) & grado(G,N,Num)), !.
```

En este caso «N» será el número de entidades que cumplen la expresión «P». Por lo tanto la Forma Lógica estará compuesta de un predicado «num_cardinal» que, en el momento de la evaluación, obtendrá «N» para que «grado» lo compare con la constante numérica «Num».

3.4. TRATAMIENTO DE LA COMPARACIÓN ENTRE UN ELEMENTO Y EL RESTO.

Este tipo de comparativas tiene un tratamiento muy distinto. Aquí no hay que realizar una comparación entre dos elementos concretos, sino que hay que comparar un elemento con el resto de elementos. Es fundamental conocer que conjunto de elementos forman el resto. Esta es una información que no viene suministrada por la Forma Lógica asociada a la comparación. Por lo tanto una tarea importante será la de buscar aquella expresión que nos determine que elementos formarán «el resto», y a la que llamaremos «ámbito del resto».

En el siguiente ejemplo:

Cual es el río gallego mas largo ?

podemos apreciar que estamos comparando en longitud un río gallego con el resto de ríos gallegos, por lo tanto el ámbito del resto estará formado por todos aquellos ríos que cumplan la propiedad de ser gallegos. El proceso de buscar el ámbito del resto se ha realizado durante el preproceso de la Forma Lógica resultante del Análisis Semántico.

La traducción de la comparación vendrá dada por una expresión que instancie un elemento y compruebe que no haya un elemento del mismo tipo perteneciente al ámbito del resto que, dependiendo del Grado de la comparación, esté más o menos cualificado, según la Base de la comparación, que el elemento que habíamos instanciado.

El código para tratar las comparaciones con el resto se presenta a continuación en el mismo orden en que se clasificó anteriormente:

(d). ENTIDAD CON EL RESTO:

- Para el subgrupo d.1 cuya Forma Lógica es comp(G,B,X,resto) donde G es adverbio de intensidad, B es adjetivo, y X es una entidad, tenemos el siguiente código de transformación,

```
qt(comp(G,B,X,Y),Cxto,Amb,FL):-
        Y==resto,
        tipo_var(X,rio,Cxto),
        cambiar(Amb,[Resto:X],AmbR),
    append(Cxto,[Resto:rio],Cxto2),
    !, qtrans(AmbR,Cxto2,_,AmbR2),
        cambia_grado(G,B,G1),
    ((elemento(B,[grande,largo,pequenyo,corto])) ->
          FL=(rio(X,_,L,_,_,_,_)&\+(rio(Resto,_,L1,_,_,_,_)&AmbR2&grado(G1,L1,L)))
    ;
          ((B=caudaloso) ->
                                                    FL=(rio(X,C,_,_,_,_,_)        &
    \+(rio(Resto,C1,_,_,_,_,_)&AmbR2&grado(G1,C1,C))))),
          !.
```

El código presentado es para el caso en que la entidad a comparar sea de tipo río, para el resto de entidades el código sería análogo. La Forma Lógica Traducida estará formada por una tupla de la entidad del tipo a comparar con el objeto de instanciar el valor del atributo determinado por la entidad que se compara, y una expresión que compruebe que no existe otro atributo asociado a alguna otra entidad («Resto») del mismo tipo perteneciente al ámbito del resto, que dependiendo del Grado de la comparación sea mayor o menor que el valor del

atributo de la entidad que habíamos instanciado. Cabe destacar que la expresión que determina el ámbito del resto viene siempre en función de la primera variable que se compara (X), por lo que deberemos sustituir cualquier ocurrencia de «X» por la variable «Resto».

- Para el subgrupo d.2 cuya Forma Lógica es comp(G,num(Z,P,N),X,resto) donde Z es entidad, el código correspondiente será:

```
qt(comp(G,num(Z,P,N),X,Y),Cxto,Amb,FL):-
  Y==resto,
  es_entidad(X,Cxto),
  es_entidad(Z,Cxto),
  !, qtrans(P,Cxto,_,P1),
  cambiar(P,[Zeta:Z,Resto:X],Q),
      busca_tipo(X,Tipoe,Cxto),busca_tipo(Z,Tipoa,Cxto),
      append(Cxto,[Resto:Tipoe,Zeta:Tipoa],Cxto2),
  cambiar(Amb,[Resto:X],AmbR),
  !, qtrans(Q,Cxto2,_,Q1),
  !, qtrans(AmbR,Cxto2,_,AmbR2),
  FL=(num_cardinal(Z,P1,N)
  &\+(AmbR2&num_cardinal(Zeta,Q1,N1)&grado(G,N1,N))),
  !.
```

En este caso vamos a comparar la propiedad de una entidad con el resto de entidades dentro del mismo ámbito. Tras realizar la comprobación de tipos, construir la expresión que determina la propiedad a comparar en función de la variable «Resto», y poner la expresión que determina el ámbito del resto en función de la variable «Resto», construiremos la FLT. Esta estará formada por el predicado «num_cardinal», el cual determinará el número de entidades «Z's» que cumplen el predicado «P» para la entidad «X», seguida de una expresión que compruebe que no haya otro elemento («Resto») del tipo de la entidad que se compara perteneciente al ámbito del resto, que esté más o menos cualificado, dependiendo del Grado de la comparación, que la entidad «X» para la propiedad que se compara.

Si en la Forma Lógica comp(G,num(Z,P,N),X,resto) Z es atributo entonces el código para el proceso «qt» será el siguiente:

```
qt(comp(G,num(Z,P,N),X,Y),Cxto,Amb,FL):-
      Y==resto,
      es_atributo(Z,Cxto),
      es_entidad(X,Cxto),
  !, qtrans(P,Cxto,_,P1),
      cambiar(P,[Zeta:Z,Resto:X],Q),
      busca_tipo(X,Tipoe,Cxto),busca_tipo(Z,Tipoa,Cxto),
      append(Cxto,[Resto:Tipoe,Zeta:Tipoa],Cxto2),
  cambiar(Amb,[Resto:X],AmbR),
      !, qtrans(Q,Cxto2,_,Q1),
```

```
!, qtrans(AmbR,Cxto2,_,AmbR2),
FL=(P1 & N=Z & \+(AmbR2 & Q1 & N1=Zeta & grado(G,N1,N))),
!.
```

En este caso, «N» será el valor del atributo «Z» perteneciente a la entidad «X». Tras realizar las comprobaciones de tipos, construir en función de la variable «Resto» la expresión que determina la propiedad a comparar, y poner la expresión que determina el ámbito del resto en función de la variable «Resto», pasaremos a construir la FLT. Esta vendrá compuesta por la expresión «P_traducido» con el objeto de instanciar «Z» al valor del atributo asociado a la entidad «X», seguida por la expresión que compruebe que no haya ningún otro valor de atributo asociado a una entidad («Resto»), del mismo tipo que «X», que pertenezca al ámbito del resto, que dependiendo del Grado de la comparación, sea mayor o menor que el valor asociado a «Z».

(e). ATRIBUTO CON EL RESTO:

- Para este grupo la Forma Lógica es comp(G,B,X,resto) donde G es el adverbio de intensidad, B es un adjetivo y X es un atributo. El código correspondiente a la transformación será de la forma:

```
qt(comp(G,B,X,Y),Cxto,Amb,FL):-
  Y==resto,
  es_atributo(X,Cxto),
  busca_tipo(X,Tipoa,Cxto),
  cambiar(Amb,[Resto:X],AmbR),
  append(Cxto,[Resto:Tipoa],Cxto2),
  !, qtrans(AmbR,Cxto2,_,AmbR1),
  cambia_grado(G,B,G1),
  ((Tipoa=caudal) ->
    (Ant=rio(_,X,_,_,_,_), AntR=rio(_,Resto,_,_,_,_),
     quitar_atomos(AmbR1,rio,AmbR2)) ;
    ((Tipoa=km) ->
      (Ant=rio(_,_,X,_,_,_), AntR=rio(_,_,Resto,_,_,_),
       quitar_atomos(AmbR1,rio,AmbR2)) ;
       ((Tipoa=km2) ->
         (Ant=com_aut(_,X,_), AntR=com_aut(_,Resto,_),
          quitar_atomos(AmbR1,aut,AmbR2)) ;
          ((Tipoa=habit) ->
            (Ant=ciudad(_,X,_), AntR=ciudad(_,Resto,_),
             quitar_atomos(AmbR1,ciudad,AmbR2)))))),
  FL=(Ant & \+(AntR2 & AmbR2 & grado(G1,Resto,X))),
  !.
```

Como siempre, tras las comprobaciones de tipos, y poner la expresión que determina el ámbito del resto en función de la variable «Resto», construiremos la FLT. Esta vendrá

formada por una tupla de la Base de Datos, con el objeto de instanciar «X» al valor de un atributo, seguido de una expresión que compruebe que no exista otro valor para un atributo del mismo tipo que «X» que sea mayor o menor, según el Grado, que el valor del atributo para el cual se instanció «X».

Cuando en la Forma Lógica asociada a la consulta tengamos que la entidad a la cual pertenece el atributo «X» sea un átomo, tendremos que el Ambito del resto estará en función de dicho átomo. Como es obvio tendremos que sustituir ese átomo por cualquier variable nueva (con la ayuda de la regla «quitar_atomos») para conseguir una correcta transformación de la Forma Lógica.

3.5. GRADO DE DEPENDENCIA RESPECTO A LA BASE DE DATOS.

Es evidente que el Analizador Pragmático está íntimamente ligado a la Base de Datos a la cual van dirigidas las consultas. Cada Base de Datos estará formada por unas tuplas muy específicas, por lo tanto, cada forma lógica parcial se transformará a otra forma lógica parcial en términos de la B. D. dependiendo de ésta. En cambio, la estructura del Analizador Pragmático será siempre la misma, es decir, todos los procesos que realiza serán transportables a cualquier B. D.

Por tanto, diremos que el proceso de adaptación de un Analizador Pragmático a una Base de Datos específica requiere unicamente introducir adecuadamente las nuevas estructuras de datos, respetando el esquema de procesos que el analizador realiza.

Centrándonos en las reglas que producen las tranformaciones de la comparación podemos ver que, hay una concreta clasificación de las formas lógicas asociadas a la comparación. En relación a la estructura de datos, eran los elementos etiquetados como «entidad» o «atributo» los que nos hacían optar por un tratamiento u otro. Por lo tanto, la fase de adaptación de una nueva Base de Datos a los procesos que tratan la comparación en el analizador pragmático, requiere solamente una perfecta distinción de los elementos que constituyen la nueva B. D., para que dependiendo de ésta puedan ser sometidos al tratamiento correspondiente.

Concluiremos diciendo que el Analizador Pragmático es dependiente de la Base de Datos en cuanto a las estructuras de datos que los procesos generales del analizador debe manipular.

4. ESTUDIO DE TIEMPOS.

Analizando los tiempos de una muestra de 150 frases, hemos obtenido una serie de resultados que presentamos a continuación.

El **tiempo medio del procesamiento completo** de una consulta es de **2'8916 segundos**, el cual se descompone de la siguiente forma:

Tiempo medio del A. Léxico ..... 0'17094 seg.
Tiempo medio del A. Sintáctico .... 1'65824 seg.
Tiempo medio del A. Semántico ..... 0'48946 seg.
Tiempo medio del A. Pragmático .... 0'57295 seg.

A partir de tres muestras de 50 frases: la primera formada únicamente por frases que no contienen ninguna comparación, la segunda solamente contiene frases con comparaciones entre dos elementos, y la tercera muestra está formada sólo por oraciones con comparaciones entre un elemento con el resto; los tiempos obtenidos son los siguientes:

Tiempo medio Muestra 1 ..... 2'11560 seg.

A. Léxico ..... 0'14000 seg.

A. Sintáctico ..... 1'38679 seg.

A. Semántico ..... 0'35800 seg.

A. Pragmático ..... 0'22040 seg.

Tiempo medio Muestra 2 ..... 4'21249 seg.

A. Léxico ..... 0'24666 seg.

A. Sintáctico ..... 2'53791 seg.

A. Semántico ..... 0'71625 seg.

A. Pragmático ..... 0'71250 seg.

Tiempo medio Muestra 3 ..... 2'39960 seg.

A. Léxico ..... 0'12680 seg.

A. Sintáctico ..... 1'08520 seg.

A. Semántico ..... 0'40320 seg.

A. Pragmático ..... 0'78400 seg.

Podemos ver que el tiempo medio de la segunda muestra, es decir, de las frases que incluyen comparaciones entre dos elementos, es bastante mas grande que el de las otras muestras. Dentro de esta muestra, hemos obtenido tiempos del orden de 20 segundos, estos tiempos tan elevados vienen asociados a frases que contienen un relativo a continuación de la comparación (ej. «Cuales son los rios con mas de 500 metros_cúbicos_por_segundo que nacen en una comunidad costera ?») Esta diferencia de tiempo se debe principalmente al análisis sintáctico.

5. CONCLUSIONES.

En este trabajo presentamos un método y una implementación para abordar el procesamiento de un subconjunto amplio de oraciones del castellano en las que se establece una comparación a través de un adjetivo o de un sustantivo.

El procesamiento de los comparativos incluye diferentes etapas. Las tres primeras corresponden al Análisis Léxico, Sintáctico y Semántico, las cuales son independientes del ámbito de aplicación. En cambio la última etapa de procesamiento, correspondiente al Análisis Pragmático es dependiente de la aplicación. En este caso hemos optado por aplicarlo a una Base de Datos Geográfica y aunque el método es válido para cualquier aplicación, el código es específico para ésta.

Con respecto a los tiempos necesarios para el procesamiento completo de las frases, hemos podido observar que éstos son bastante aceptables, recayendo el peso fundamentalmente en los tiempos consumidos en la etapa de Análisis Sintáctico.

BIBLIOGRAFÍA.

[ABRAMS, 89] Abramson, H. y Dahl, V.(1989)
Logic Grammars,
Symbolic Computation Series.
Springer-Verlag.

[ALLEN, 87] Allen,J. (1987)
Natural Language Understanding.
Benjaming Cummings series in Computer Science.

[GAZDAR, 89] Gazdar,G.y Mellish,C. (1989)
Natural Language Processing in Prolog.
Addison-Wesley.

[MCCORD, 85] McCord,M. (1985)
Modular Logic Grammars.
Proc. Association of Computational Linguistics.

[MCCORD, 82] McCord,M. (1982)
Using Slots and Modifiers in Logic Grammars for Natural Language.
Artificial Intelligence, vol. 18

[MORENO,92a] Moreno, L. y Palomar, M.(1992)
Semantic Constraints in a Syntactic Parser.
en Database and Expert Systems Applications
Springer-Verlag (pendiente de publicación)

[MORENO, 92b] Moreno, L.,Molina, A. y Pastor M.(1992)
Interpretación de la comparación a través de la Lógica.
Informe Interno (DSIC-Univ. Politéc. de Valencia)

[PALOMA, 91] Palomar, M. , Moreno, L. y Pascual, A. (1991)
Semantic Interpretation of Natural Language in PROLOG:Logical Forms.
Proc. Database and Expert Systems Applications
Karagiannis, D. Ed. (3-211-82301-8)
Springer-Verlag

[PALOMA, 90] Palomar, M. y Moreno, L. (1990)
Syntactic and Morphologic Analysis of Natural Language Expressions.
Proc. Artificial Intelligence Application & Neural Networks
Hamza, M.H. Ed. (0-88986-152-8)
Acta Press

[VANCAN, 86] Van Caneghem,M y Warren,D eds. (1986)
Logic Programming and its applications.
Ablex Publishing Coorporation.

[WALKER, 89] Walker,A y otros eds. (1989)
Knowledge Systems and Prolog.
Addison-Wesley.

ANEXO I. EJEMPLOS DE FRASES COMPARATIVAS

En este anexo se presentan diversos ejemplos de consultas a una Base de Datos Geográfica, cuyo esquema lógico es el siguiente:

mar_com(NOMCOM, NOMMAR).
rio(NOMRIO, CAUDAL, LONGITUD, NOMCOM_NACE,
NOMCOM_DESEM,NOMSIST_M_NACE, NOMMAR_DESEM).
com_aut(NOMCOM, EXTENSION, NOMCIU_CAP).
pasar(NOMCOM, NOMRIO).
ciudad(NOMCIU, NOHABIT, NOMCOM).
sist_m(NOMSIST-M).
com_sist_m(NOMCOM, NOMSIST_M).
mar(NOMMAR).

Cada ejemplo se compone de:

a). frase introducida por el usuario
b). forma lógica generada por el módulo semántico
c). forma lógica tras el preproceso
d). forma lógica en términos de la Base de Datos
e). la respuesta.

EJEMPLO 1.

a)que rios son mas grandes que el rio Turia ?
b)preg(_1031,rio(_1031)&ex(rio(_1916)&_1916 = turia,comp(mas,grande,_1031,_1916)))
c)preg(_1031,ex(comp(mas,grande,_1031,turia)))
d)preg(_1031,rio(_1031,_6379,_6363,_6381,_6382,_6383,_6384) &
rio(turia,_6390,_6362,_6392,_6393,_6394,_6395) & grado(mas,_6363,_6362))
e)Ebro, Minyo, Segura, Pisuerga, Guadalquivir, Esla, Tormes, Tajo, Guadiana, Genil, Duero, Jucar y Segre.

EJEMPLO 2.

a)Que rio gallego es mas grande ?
b)preg(_800,(rio(_800) & gallego(_800)) & comp(mas,grande,_800,resto))
c)preg(_800,gallego(_800) & comp(mas,grande,_800,resto))
d)preg(_800,pasar(galicia,_800) & rio(_800,_4693,_4677,_4695,_4696,_4697,_4698) &
\+(rio(_4577,_4706,_4676,_4708,_4709,_4710,_4711) & pasar(galicia,_4577) &
grado(mas,_4676,_4677)))
e)Minyo.

EJEMPLO 3.

a)Que autonomias son mas extensas que la Comunidad_valenciana ?

b)preg(_1271,aut(_1271) & comp(mas,extenso,_1271,valencia))

c)preg(_1271,comp(mas,extenso,_1271,valencia))

d)preg(_1271,com_aut(_1271,_4938,_4939) & com_aut(valencia,_4945,_4946) & grado(mas,_4938,_4945))

e)Andalucia, Castilla_leon, Catalunya, Castilla_la_mancha, Galicia, Aragon y Extremadura.

EJEMPLO 4.

a)Que comunidad costera es mas extensa ?

b)preg(_898,(aut(_898) & costero(_898)) & comp(mas,extenso,_898,resto))

c)preg(_898,costero(_898) & comp(mas,extenso,_898,resto))

d)preg(_898,mar_com(_898,_5051) & com_aut(_898,_5189,_5190) & \+(com_aut(_5094,_5198,_5199) & mar_com(_5094,_5171) & grado(mas,_5198,_5189)))

e)Andalucia.

EJEMPLO 5.

a)Dime los rios de longitud mas grande que la longitud que tiene el rio Turia.

b)preg(_1538,ex(km(_2345)&ex(rio(_2839)&_2839=turia,tener1(_2839,_2345)), ex(km(_1931) & comp(mas,grande,_1931,_2345),rio(_1538) & de(_1931,_1538))))

c)preg(_1538,ex(ex(tener1(turia,_2345)), ex(comp(mas,grande,_1931,_2345),de(_1931,_1538))))

d)preg(_1538,rio(turia,_12269,_2345,_12271,_12272,_12273,_12274) & (rio(_12392,_12393,_1931,_12395,_12396,_12397,_12398) & grado(mas,_1931,_2345) & rio(_1538,_12415,_1931,_12417,_12418,_12419,_12420))

e)Ebro, Minyo, Segura, Pisuerga, Guadalquivir, Esla, Tormes, Tajo, Guadiana, Genil, Duero, Jucar y Segre.

EJEMPLO 6.

a)Dime los rios de longitud mayor que 200 kilometros.

b)preg(_1049,num(_1855,km(_1855)&ex(km(_1442)&comp(mas,grande,_1442,_1855), rio(_1049) & de(_1442,_1049)),200))

c)preg(_1049,num(_1855,ex(comp(mas,grande,_1442,_1855),de(_1442,_1049)),200))

d)preg(_1049,_1855 = 200 & (rio(_8338,_8339,_1442,_8341,_8342,_8343,_8344) & grado(mas,_1442,_1855)) & rio(_1049,_8361,_1442,_8363,_8364,_8365,_8366))

e)Sil, Ebro, Minyo, Segura, Pisuerga, Turia, Guadalquivir, Esla, Tormes, Tajo, Guadiana, Genil, Duero, Cabriel, Jucar, Ter, Alagon, Gallego, Segre, Jalon y Guadajoz.

EJEMPLO 7.

a)Dime el rio valenciano de longitud mayor.

b)preg(_863,ex(km(_1341)&comp(mas,grande,_1341,resto),(rio(_863)&valenciano(_863)) & de(_1341,_863)))

c)preg(_863,ex(comp(mas,grande,_1341,resto),valenciano(_863) & de(_1341,_863)))

d)preg(_863,(rio(_7639,_7640,_1341,_7642,_7643,_7644,_7645) &
\+(rio(_7648,_7649,_7360,_7651,_7652,_7653,_7654) &
(rio(_863,_7557,_7360,_7559,_7560,_7561,_7562) & pasar(valencia,_863)) &
grado(mas,_7360,_1341))) &
pasar(valencia,_863) & rio(_863,_7919,_1341,_7921,_7922,_7923,_7924))

e)Jucar.