INCORPORAR RESTRICCIONES SEMANTICAS EN EL ANALISIS SINTACTICO: IRSAS

L. Moreno / F. Andrés/ M. Palomar

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia

1. INTRODUCCIÓN.

En la fase de análisis sintáctico de un Lenguaje Natural, frecuentemente, nos encontramos con la necesidad de incorporar información semántica para resolver cierto tipo de ambigüedades léxicas.

Cuando un objeto(palabra o agrupación de éstas) tiene múltiples significados, deberíamos distinguir de entre éstos cual es el mas adecuado, en un momento dado, por su contexto semántico. Una posibilidad sería introducir un rasgo específico para cada significado del objeto.

En el proceso de desambiguación de las palabras se tiene en cuenta las otras palabras que le acompañan en la oración, jugando un papel crucial los verbos para desambiguar los sustantivos. De la misma manera, las palabras que actúan como sujeto o complemento de un verbo son potentes desambiguadores de éste.

El hecho de que una palabra restrinja el significado de las otras, conocido por **Restricción Seleccional (RS)**, se representa como un patrón a emparejar con alguna estructura sintáctica. Las restricciones seleccionales pueden aparecer también en los adjetivos, sustantivos y preposiciones.

Necesitamos pues desarrollar un mecanismo que nos permita extender el análisis sintáctico haciendo explícitas las restricciones seleccionales y plantearnos algún método para expresar el conocimiento sobre la estructura del mundo, a través de una jerarquía de rasgos semánticos.

2. MÉTODO IRSAS.

Previo a abordar el problema de especificación de las RS deberemos establecer como identificar a un objeto. El aspecto principal es observar la manera en que los objetos del universo se clasifican dentro de grupos por sus propiedades o rasgos semánticos, para asociar a cada objeto el conjunto de rasgos que le caracteriza. Así pues, un objeto consistirá de:

- Un **tipo** el cual es un conjunto de rasgos denotando una parte del universo al que pertenece.
- Un **referente** identificador del concepto genérico o individual.

Posteriormente se determinará (restringirá) el tipo de los potenciales constituyentes asociados a un determinado verbo, adjetivo, sustantivo o preposición al establecer las RS.

2.1. *JERARQUÍA DE RASGOS SEMÁNTICOS.*

Afrontar el problema de construir una jerarquía de rasgos, de menor a mayor especificidad, no es algo trivial, ya que las relaciones entre rasgos pueden intersectar, o ser excluyentes, o incluso algunos pueden ser la conjunción de otros tipos. Además hay que considerar que si un rasgo es un subrasgo de otro rasgo, el primero debe heredar todas las propiedades del segundo. Dahlgren y McDowell realizan un trabajo interesante sobre representación del conocimiento que exponen en [7].

En primer lugar vamos a especificar la jerarquía de rasgos mediante la sintaxis propuesta por McCord en [17]. La especificación se realizará mediante dos tipos de relaciones. El primer tipo de relaciones, a las que llamaremos **relaciones de herencia y división**, se denotan mediante la siguiente sintaxis:

A <=> B1 v B2 v..v Bn.

pudiendo leerse como "el rasgo A se divide en los rasgos B1 ó B2 ó..ó Bn". Donde la disyunción **ó** es exclusiva. Puede existir más de una sentencia de división para un mismo rasgo, lo que significa que un rasgo se puede dividir de diferentes formas(**clasificación cruzada**).

El segundo tipo de relaciones utilizadas son las llamadas **relaciones de implicación** y se introducen mediante la siguiente sintaxis:

A => C

pudiendo leerse como "el rasgo C es implicado por el rasgo A", donde A tiene un grado de especificidad mayor que C.

Un ejemplo de clasificación jerárquica u ontología de rasgos semánticos podría ser el siguiente:

**Relaciones de división y herencia ->*

1) entidad <=> concreta v abstracta.
2) entidad <=> individual v colectiva.
3) abstracta <=> estado v acción.
4) estado <=> color v forma v anímico.
5) concreta <=> viviente v no_viviente.
6) concreta <=> sólido v líquido v gas.
7) viviente <=> animal v planta.
8) animal <=> humano v no_humano.
9) animal <=> macho v hembra.

**Relaciones de implicación ->*

1) animal => sólido.
2) gas => no_viviente.

donde sobre "entidad", "concreta" y "animal" se han definido clasificaciones cruzadas.

Grafo de Herencias, Grafo de Incompatibilidades, Grafo de Implicaciones.

Las interrelaciones existentes entre los rasgos que constituyen una ontología, pueden representarse mediante tres Grafos. Vamos a definir cada uno de ellos, a partir de las relaciones de división y herencia, e implicación.

Las **relaciones de división y herencia** se pueden representar mediante dos grafos:

- Grafo de Herencias.
- Grafo de Incompatibilidades.

El **grafo de herencias** (GH) es un grafo dirigido que constituye una arborescencia, cuya raíz es el rasgo "entidad". En él se representa la manera en que los rasgos semánticos se derivan unos de los otros. Sea GH(Vh,Eh) el grafo de herencias, donde Vh y Eh son el conjunto de vértices y el conjunto de aristas respectivamente, definidos de la siguiente forma:

- Vh :"entidad" $\in$ Vh, y para toda relación de división y herencia (a <=> b1 v..v bn), se cumple que: b1,..,bn $\in$ Vh.
- Eh : para toda relación de división y herencia (a <=> b1 v..v bn), se cumple que: (a,b1),..,(a,bn) $\in$ Eh.

El **grafo de incompatibilidades** (GI) es un grafo no dirigido, en el que se representan las incompatibilidades semánticas de unos rasgos con otros. Sea GI(Vi,Ei) el grafo de incompatibilidades, donde Vi y Ei se definirán:

- Vi: Vi=Vh.
- Ei: para toda relación de división y herencia (a <=> b1 v b2 v b3 v..v bn), se cumple "(b,c) Œ Ej':

$(b,c) \in Ei.$

$\forall u \in \Gamma H^{+*}(b), (u,c) \in Ei.$

$\forall v \in \Gamma H^{+*}(c), (v,b) \in Ei.$

$\forall u \in \Gamma H^{+*}(b)$ y $\forall v \in \Gamma H^{+*}(c), (u,v) \in Ei.$

siendo $\Gamma H^{+*}(x)$ el conjunto de vértices accesibles por x, es decir, el conjunto de rasgos descendientes de x en el grafo de herencias. Y Ej' es el conjunto de pares de rasgos directamente incompatibles, construido por la regla de división y herencia j-ésima, combinando cada rasgo que aparece en la parte derecha de la regla con todos los rasgos que aparecen a la derecha de éste:

(b1,b2),(b1,b3),..,(b1,bn),

(b2,b3),..,(b2,bn),..,(bn-1,bn) $\in$ Ej'.

Examinemos ahora la información que aportan las **relaciones de implicación**, representándolas mediante un grafo(GM): Grafo de implicaciones.

Las relaciones de implicación denotan que unos rasgos pueden deducirse a partir de otros sin que entre ellos exista una relación de herencia. Aportan información del universo de discurso que no puede ser reflejada por las relaciones de división y herencia.

Siendo GH(Vh,Eh) el grafo de herencias antes definido, el **grafo de implicaciones** GM(Vm,Em) será un grafo dirigido cuyo conjunto de vértices Vm, y de aristas Em se definirán de la siguiente forma:

- Vm : Vm = Vh
- Em : para toda relación de implicación (a => b), se cumple que:

(a,b) $\in$ Em.

" u /(u $\in \Gamma H^{-*}(b)$ L u no $\in \Gamma H^{-*}(a)$), (a,u) $\in$ Em.

" v $\in \Gamma H^{+*}(a)$, (v,b) $\in$ Em.

" v $\in \Gamma H^{+*}(a)$, $\forall$ u /(u $\in \Gamma H^{-*}(b)$ L

u no $\in \Gamma H^{-*}(a)$), (v,u) $\in$ Em.

donde $\Gamma H^{-*}(x)$ es el conjunto de todos los vértices que acceden a x, es decir, todos los rasgos semánticos ancestros de x en el Grafo de herencias. Así pues, por (2) los ancestros de "b" que no sean también ancestros de "a" son implicados por "a"; por (3) "b" es implicado por los descendientes de "a"; y por (4) los ancestros de "b" que no sean también ancestros de los descendientes de "a", son implicados por los descendientes de "a".

Así en $\Gamma M^{+*}(d)$ tendríamos el conjunto de todos los rasgos semánticos deducibles según las reglas de implicación a partir del rasgo d.

Además, las relaciones de implicación permiten deducir relaciones entre rasgos semánticos que no poseen una relación de herencia, e introducen también nuevas relaciones de exclusión,

ampliando el grafo de incompatibilidades semánticas con nuevas aristas. El grafo de incompatibilidades definido anteriormente será ampliado añadiendo al conjunto de aristas Ei, un nuevo conjunto de aristas Ei' que se define a continuación :

Ei': $\forall(a,b). \in Em$, si existe una relación de división y herencia (h <=> c1 v c2 v b v..v cn), se cumple que:

$(a,c1),(a,c2),..,(a,cn) \in Ei'$.

$\forall v1 \in \Gamma H+^*(c1), (a,v1) \in Ei'$.

$\forall v2 \in \Gamma H+^*(c2), (a,v2) \in Ei'$.

...

$\forall vn \in \Gamma H+^*(cn), (a,vn) \in Ei'$.

Condiciones de verificación de una ontología.

Partiendo de los tres grafos definidos, diremos que una clasificación ontológica está definida correctamente si se verifica que :

-El grafo de herencias constituye una arborescencia: $\forall a \in Vh$ (Cardinal(ΓH-1(a))=1) ó (Cardinal(ΓH-1(a))=0 y a es raíz).

-El conjunto de vértices de los 3 grafos es el mismo:
Vh = Vi = Vm = V.

-Las relaciones de implicación no generan absurdos:

$\forall b \in V$ se verifica que

$\forall a \in V$, si $b \in \Gamma M+^*(a)$, b no $\in \Gamma I(a)$

donde ΓH-1(x) es el conjunto de rasgos que acceden directamente al rasgo x en el grafo de herencias, y $\Gamma I(x)$ es el conjunto de rasgos incompatibles con el rasgo x.

2.2. *ESTRUCTURA DE LAS ENTRADAS LÉXICAS*

Como ya se indicó con anterioridad, a un objeto se le asocia un tipo (conjunto de rasgos) y un referente(identificador individual o genérico). Esta información se registra en las entradas léxicas.

El tipo de cada objeto individual se representará mediante una lista de rasgos a la cual denominaremos **lista candidata**, cuyo referente será un término constante.

Además, si procede, se determinará el tipo de los potenciales constituyentes que le acompañarán en una oración mediante una estructura de huecos, definiendo la Restricción Seleccional correspondiente. A dichos constituyentes se les referenciará mediante un identificador genérico, término variable, junto con su tipo definido como una lista de rasgos a la que denominaremos **lista de restricciones.** Por ejemplo, la entrada léxica de un verbo tendrá la siguiente estructura:

verbo(FC, PRED, TS, MSUJ, HUECOS)

donde FC: forma de citación o verbo en infinitivo.
PRED: forma lógica asociada al verbo.
TS: tipo semántico asociado al verbo.
MSUJ: sujeto y tipo semántico asociado.
HUECOS: estructura de Huecos de los complemento del verbo.

Así pues, para el verbo **dar** podríamos tener la entrada
verbo(dar,dar(SUJ,X,Y),[accion],SUJ:[animal],
(od:X:[entidad]),(oi:Y:[concreta])])
Luego el verbo **dar** tiene por tipo semántico [accion], y aparecen tres constituyentes alrededor del verbo: el sujeto (SUJ) con tipo semántico [animal], un objeto directo (X) con tipo [entidad], y por último, un objeto indirecto (Y) con tipo semántico [concreta].

2.3. *CONSISTENCIA DE LAS LISTAS DE RASGOS.*

Partiendo de los grafos GH(V,Eh), GI(V,Ei), GM(V,Em) y suponiendo que a, b, c son rasgos semánticos definiremos los siguientes conceptos :

- a y b son **excluyentes** si (a,b) ∈ Ei.

- a se **deriva** de b si a ∈ ΓH+*(b).
- a y b **no están en relación de derivación** si a no deriva de b y b no deriva de a.

- a es **implicado** por b si a ∈ ΓM+*(b).
- a y b **no están en relación de implicación** si a no es implicado por b y b no es implicado por a.
- Llamamos **clausura** de un rasgo semántico "a", al conjunto de rasgos deducidos a partir de él("a", todos sus ancestros y todos los implicados por él):

clausura(a) = {u ∈ V /u ∈ ΓH-*(a) ó u ∈ ΓM+*(a) ó u = a}.
es decir,
clausura(a) = ΓH-*(a) U ΓM+*(a) U {a}.

Dado A={a,b,..,e}, entonces
clausura(A)=clausura(a) U clausura(b) U..U clausura(e).

- a y b son **disjuntos** si se cumple:

(a no ∈ ΓI(b)) L

(a no ∈ clausura(b)) L (b no ∈ clausura(a))

es decir, a y b son **disjuntos** si no son excluyentes, no están en relación de derivación y no están en relación de implicación.

- Dada una lista de rasgos semánticos L, decimos que ésta es **mínimo representativa** si se verifica que:

∀ a,b ∈ L,

(a no ∈ clausura(b)) L (b no ∈ clausura(a)).

Si en una lista de rasgos semánticos L aparecen rasgos en relación de derivación, podemos eliminar los rasgos más generales o menos específicos sin que la lista pierda representatividad semántica. Lo mismo ocurre cuando aparecen tipos en relación de implicación, podemos eliminar aquellos tipos que son implicados sin que se pierda contenido semántico.

Veamos un ejemplo según la jerarquía definida inicialmente:
L1=[individual, animal, viviente, solido].
L2=[individual, animal, solido].
L3=[individual, animal].
aquí L3 es mínimo representativa de las listas anteriores, ya que animal deriva de viviente e implica sólido.

Una lista de restricciones que representa las condiciones que debe cumplir un objeto candidato, **debe ser mínimo representativa** para evitar gastar tiempo de computación innecesario, probando restricciones de rasgos que son más generales que otras restricciones que ya existen en la lista.

- Sobre un rasgo "r" pueden definirse varias relaciones de herencia y división (clasificación cruzada), cada una de las cuales constituirá un conjunto de rasgos excluyentes entre si (Ri={r1, r2,...rm}). Al conjunto de rasgos formado por todos los rasgos que aparecen en la parte derecha de las relaciones que constituyen la clasificación cruzada para "r" la denotamos por R, siendo R=R1 U R2 U...Rn, teniendo que cumplir R las siguientes condiciones:

1) $\forall$ Ri, Rj , $Ri \cap Rj = \emptyset$

2) $\forall$ Rj, $\forall$ ri $\in$ Rj y $\forall$ rk $\in$ Rj , ri y rk son excluyentes siendo $i \neq k$.

3) $\forall$ Ri, $\forall$ Rj /($i \neq j$), $\forall$ ri $\in$ Ri, $\forall$ rj $\in$ Rj, ri y rj no son excluyentes.

- Dado una lista de rasgos semánticos L decimos que ésta es **completa** cuando $\forall$ d $\in$ clausura (L) se cumple que:

si d $\in$ Ri, $\forall$ Rj $\exists$ b /((b $\in$ clausura (L)) L (b $\in$ Rj)),es decir, L es completa si para todo rasgo d en clausura de L se verifica que si ese rasgo d se deriva directamente de otro rasgo c (d es hijo de c) del cual se ha definido una clasificación cruzada entonces en dicha clausura(L) se a de encontrar al menos un rasgo de cada una de las otras relaciones de división y herencia de la clasificación cruzada.

Según la clasificación ontológica inicial los siguientes ejemplos son listas completas :

[entidad]. [concreta,colectiva].
[viviente,sólido,individual].
[animal,sólido,colectivo]. [animal,colectivo].
[humano,hembra,macho,individual].

De todo lo anterior, deducimos, que **una lista candidata debe ser siempre completa**. A una lista de restricciones, no se le exige la condición de completa, porque no representa propiedades semánticas de entidades sobre nuestro universo de discurso, sino solo condiciones a verificar.

2.4. *COMPATIBILIDAD ENTRE LISTA CANDIDATA Y LISTA DE RESTRICCIONES*

Una lista de rasgos semánticos L, decimos que cumple las restricciones impuestas por un rasgo semántico "b" si b e clausura(L), lo que significa que b es deducido a partir de L.

Una **lista candidata LC**, es **compatible** o cumple las restricciones impuestas por una **lista de restricciones LR**, cuando se verifique que:

$\forall$ b $\in$ LR , $\exists$ a / ((a $\in$ LC) L (b $\in$ clausura(a)),

es decir, LR esta incluida en clausura(LC). La lista de restricciones se deduce a partir de la lista candidata.

A este mecanismo de verificación le denominaremos **condiciones de emparejamiento**.

3. IMPLEMENTACIÓN DEL MÉTODO.

El método IRSAS se ha implementado en Arity PROLOG. A continuación se expone la representación interna de la jerarquía de rasgos y una serie de algoritmos que nos permitirán implementar los conceptos definidos en el punto anterior a partir de dicha representación.

3.1. *IMPLEMENTACIÓN DE LA JERARQUÍA DE RASGOS.*

Las relaciones de división y herencia están representadas por los axiomas ontolog(a,Lh), y las relaciones de implicación por los axiomas ontolog2(a,Li). Los argumentos en ambos se construirán de la siguiente forma:

Sea R el conjunto de rasgos semánticos pertenecientes a una ontología.

$\forall$ a $\in$ R /(a <=> b1 v b2 v b3 v ...
a <=> c1 v c2 v c3 v ...
...),
$\exists$ ontolog(a,[[b1,b2,b3,..],[c1,c2,c3,..],..]).

$\forall$ a $\in$ R / (a => b1
a => b2
...),
$\exists$ ontolog2(a,[b1,b2,..]).

3.2. *ALGORITMOS.*

Si observamos las definiciones de los conceptos teóricos expuestas anteriormente (clausura, lista completa, lista mínimo representativa, condiciones de emparejamiento...), comprobaremos que todas utilizan algunas de las siguientes funciones :

- ΓH-1(a).
- ΓH-*(a).
- ΓM+*(a).
- ΓI(a).

Por lo tanto, si desarrollamos unos algoritmos que nos permitan obtener a partir de ontolog(a,Lh) y ontolog2(a,Li) las funciones anteriores, obtendremos la forma de implementar dichos conceptos.

Rasgo padre de "a": ΓH-1(a).

ΓH-1(a)= { b } si $\exists$ ontolog(b,[[c1,a,c2,..]..]).
ΓH-1(a) = ø si no $\exists$ ontolog(b,[[c1,a,c2,..]..]).
(cuando ΓH-1(a) sea ø, a es raíz de la arborescencia)

Rasgos ancestros de "a": GH-*(a).

ΓH-2(a) = ΓH-1(ΓH-1(a)).
ΓH-3(a) = ΓH-1(ΓH-2(a)).
...
ΓH-n(a) = ΓH-1(ΓH-n-1(a)).

$\Gamma H^{-*}(a) = \Gamma H^{-1}(a) \cup \Gamma H^{-2}(a) \cup .. \cup \Gamma H^{-n}(a)$,
siendo $\Gamma H^{-n}(a) = \emptyset$.

Si B = {b1,b2,b3,..} ,
$\Gamma H^{-*}(B) = \Gamma H^{-*}(b1) \cup \Gamma H^{-*}(b2) \cup \Gamma H^{-*}(b3) \cup ...$

Rasgos implicados por "a": $GM^{+*}(a)$.

Definiremos primero $\Gamma M'^{+1}(a)$ y $\Gamma M^{+1}(a)$.

->Rasgos directamente implicados por "a": $\Gamma M'^{+1}(a)$.

$\Gamma M'^{+1}(a) = \{b1,b2,..\}$ si $\exists$ ontolog2(a,[b1,b2,..]).
$\Gamma M'^{+1}(a) = \emptyset$ si no $\exists$ ontolog2(a,[b1,b2,..]).

Si C={c1,c2,..},
$\Gamma M'^{+1}(C) = \Gamma M'^{+1}(c1) \cup \Gamma M'^{+1}(c2) \cup ...$

->Rasgos directamente implicados por "a" y los ancestros de éstos: $\Gamma M^{+1}(a)$.

$\Gamma M^{+1}(a) = \Gamma M'^{+1}(a) \cup \Gamma H^{-*}(\Gamma M'^{+1}(a))$.

Si D={d1,d1,..},
$\Gamma M^{+1}(D) = \Gamma M^{+1}(d1) \cup \Gamma M^{+1}(d2) \cup ...$
Por lo tanto,
$\Gamma M'^{+2}(a) = \Gamma M'^{+1}(\Gamma M'^{+1}(a))$.
$\Gamma M^{+2}(a) = \Gamma M'^{+2}(a) \cup \Gamma H^{-*}(\Gamma M'^{+2}(a))$. (*)
...
$\Gamma M'^{+n}(a) = \Gamma M'^{+1}(\Gamma M'^{+n-1}(a))$.
$\Gamma M^{+n}(a) = \Gamma M'^{+n}(a) \cup \Gamma H^{-*}(\Gamma M'^{+n}(a))$.
$\Gamma M^{+*}(a) = \Gamma M^{+1}(a) \cup \Gamma M^{+2}(a) \cup .. \cup \Gamma M^{+n}(a)$,
siendo $\Gamma M^{+n}(a) = \emptyset$.

(*$\Gamma M^{+2}(a)$ representa los implicados de los implicados por a, los implicados por los ancestros de los implicados, más los ancestros de todos los anteriores).

Rasgos incompatibles con "a": $\Gamma I(a)$.

Definiremos primero $\Gamma H^{+1}(a)$, $\Gamma H^{+*}(a)$ y $\Gamma T'(a)$.

->Rasgos hijos de "a":$\Gamma H^{+1}(a)$.

$\Gamma H^{+1}(a)$={b1,b2,..,c1,c2,...}
si $\exists$ ontolog(a,[b1,b2,..],[c1,c2,..],..]).
$\Gamma H^{+1}(a) = \emptyset$
si no $\exists$ ontolog(a,[b1,b2,..],[c1,c2,..],..]).
($\Gamma H^{+1}(a)$ es $\emptyset$ cuando "a" es hoja de la arborescencia).

Si B={b1,b2,..},
$\Gamma H+1(B)=\Gamma H+1(b1) \cup \Gamma H+1(b2) \cup ...$

->Rasgos descendientes de "a": $\Gamma H+^*(a)$.

$\Gamma H+2(a) = \Gamma H+1(\Gamma H+1(a))$.
$\Gamma H+3(a) = \Gamma H+1(\Gamma H+2(a))$.
..
$\Gamma H+n(a) = \Gamma H+1(\Gamma H+n-1(a))$.
$\Gamma H+^*(a) = \Gamma H+1(a) \cup \Gamma H+2(a) \cup .. \cup \Gamma H+n(a)$,
siendo $\Gamma H+n(a)=\emptyset$.

Si C={c1,c2,..},
$\Gamma H+^*(C)=\Gamma H+^*(c1) \cup \Gamma H+^*(c2) \cup ...$

->Rasgos directamente incompatibles con "a": $\Gamma\Gamma'(a)$.

$\Gamma\Gamma'(a) = B \cup \Gamma H+^*(B)$
si $\exists$ ontolog(c,[[b1,a,b2,..] / B={b1,b2,..}.
$\Gamma\Gamma'(a) = \emptyset$ si no $\exists$ ontolog(c,[[b1,a,b2,..]..]).
(Si un rasgo es incompatible con otro también lo es con sus descendientes).

Si D={d1,d2,..}, entonces $\Gamma\Gamma'(D)=\Gamma\Gamma'(d1) \cup \Gamma\Gamma'(d2) \cup ..$
Por lo tanto,
$\Gamma I(a) = d1(a) \cup d2(a)$
donde,
$d1(a)=\Gamma\Gamma'(a) \cup \Gamma\Gamma'(\Gamma H-1(a)) \cup \Gamma\Gamma'(\Gamma H-2(a)) \cup .. \cup \Gamma\Gamma'(\Gamma H-n(a))$,
siendo $\Gamma\Gamma'(\Gamma H-n(a)) = \emptyset$.

podemos escribirlo como,
$d1(a) = \Gamma\Gamma'(a) \cup \Gamma\Gamma'(\Gamma H-^*(a))$
es decir, un rasgo es incompatible con los rasgos incompatibles de sus ancestros, y además,
$d2(a)=\Gamma\Gamma'(a) \cup \Gamma\Gamma'(\Gamma M+1(a)) \cup \Gamma H'(\Gamma H-2(a)) \cup .. \cup \Gamma\Gamma'(\Gamma M+n(a))$,

siendo $\Gamma\Gamma'(\Gamma M+n(a)) = \emptyset$, podemos escribirlo como,
$d2(a) = \Gamma\Gamma'(a) \cup \Gamma\Gamma'(\Gamma M+^*(a))$
es decir, un rasgo es incompatible con los rasgos incompatibles de sus implicados. Así pues,
$\Gamma I(a) = d1(a) \cup d2(a)$.
$\Gamma I(a) = \Gamma\Gamma'(a) \cup \Gamma\Gamma'(\Gamma H-^*(a)) \cup \Gamma\Gamma'(\Gamma M+^*(a))$.
$\mathbf{\Gamma I(a)} = \Gamma\Gamma'(clausura(a))$.

Un rasgo es incompatible con aquellos rasgos que son incompatibles con los rasgos deducidos de él.

Utilizando estas definiciones algorítmicas, en el sistema IRSAS se encuentran desarrollados en Arity PROLOG una serie de predicados que implementan las definiciones teóricas expuestas.

La cláusula que ejecuta la verificación correcta de una ontología, comprueba las tres condiciones que debe cumplir una ontología que este definida correctamente. La última condición ("las relaciones de implicación no generan absurdos"), se ha redefinido utilizando el algoritmo de la clausura:

$\forall x \in V$;

si $x \in$ clausura($\Gamma M'+1(a)$) entonces x no $\in \Gamma I(a)$.

donde el algoritmo asociado a la condición anterior será el siguiente:

" ontolog2(a,[b1,b2,..,b3]) / B={b1,b2,..,bn}, se verifica que:

$\forall x \in$ clausura(B), x no $\in \Gamma T'$(clausura(a)).

4. ESTUDIO COMPARATIVO.

Se ha implementado el **sistema IRSAS** y otro sistema basado en la estructura denominada **árbol-tipo**, propuesto por McCord en [11] y [17].

Los dos sistemas tienen el mismo objetivo: poder comprobar las restricciones semánticas en el proceso de Análisis Sintáctico de una oración del Lenguaje Natural. En ambos casos se representa el tipo asociado a un objeto mediante una lista de rasgos.

Como expresa Beringer en [5], PROLOG tiene limitaciones para detectar incompatibilidades entre dos representaciones: "Dos representaciones equivalentes de un mismo objeto, pero sintácticamente diferentes no pueden unificarse". Por ejemplo, conjunto(a.b.c.nil) y conjunto(c.b.a.nil) deben interpretarse como el conjunto {a,b,c}, pero PROLOG no es capaz de emparejar ambos conjuntos.

La diferencia fundamental entre los sistemas implementados radica en el mecanismo que se propone para verificar que dos listas de rasgos (candidata y de restricciones) son compatibles. McCord propone transformar las listas de rasgos en arboles-tipo y luego unificarlas, mientras que IRSAS establece el mecanismo denominado condiciones de emparejamiento.

Después de introducir diversas mejoras en el sistema de arboles-tipo (como la poda del árbol por aquellos nodos que representen tipos atómicos en la lista de restricciones antes de la unificación), hemos realizado un estudio comparativo entre este método y el propuesto. Primero mostraremos el análisis comparativo técnico de complejidad de ambos métodos (a través del estudio de la cota superior) y la diferencia de tiempos consumidos por ambos, en el proceso de comprobación semántica durante la fase de análisis sintáctico, a través de unos ejemplos.

Análisis comparativo de complejidad :

(a) *Condiciones de emparejamiento.*

$t_{compatib}(n, m) \in O(n \times m)$

donde :

n = talla de la lista de Restricciones.
m = talla de la lista Candidata.

(b) *Arboles-tipo.*

$t_{compatib}(n, m, p) \in O(n \times m \times p)$.

donde :

n = talla de la lista de Restricciones.

m = talla de la lista Candidata.
p = nº nodos de los subárboles.

siendo **compatib** la cláusula que ejecuta el proceso de verificar la compatibilidad entre dos listas de rasgos.

Tabla de tiempos en comprobación semántica:

Como ejemplo tomaremos un conjunto de oraciones sintácticamente correctas:

1.- El río Turia pasa por la ciudad de Valencia.
2.- ¿Cuál es la extension de la provincia de Cuenca?
3.- El padre del abogado de Juan cree en la ambición del hombre.
4.- Ella amó siempre a Pedro con fuerza.
5.- ¿Cuál es el caudal del río Ebro en Zaragoza?
6.- El médico era visto por la mujer que daba agua a un cachorro.
7.- Todo hombre espera amar a una mujer que crea en él.
8.- Algunos abogados quieren dar mayor abrigo a la ciudad.
9.- María poda las tapas del libro con unas tijeras.
10.-El reloj talla las horas que nacen en él.

Si para cada una de ellas registramos el tiempo promedio consumido en realizar el proceso **compatib**, podremos confeccionar la siguiente tabla:

Oración	Condiciones de emparej.	Arboles-tipo	Diferencia
1	1.58 seg.	2.16 seg.	0.58 seg.
2	2.05	3.17	1.12
3	1.36	1.95	0.59
4	1.20	1.25	0.05
5	1.55	2.50	0.95
6	1.70	2.60	0.80
7	1.34	2.68	1.34
8	2.78	2.93	0.15
9	0.24	0.25	0.01
10	0.16	0.19	0.03

Como puede observarse en la tabla de tiempos las diferencias de tiempos para cada una de las oraciones fluctúa entre 0.01 segundo (Oración 9) y 1.34 segundos (Oración 7). Esta diferencia se debe a la cantidad de comprobaciones de compatibilidad entre listas de rasgos de las diferentes Restricciones Seleccionales de los objetos que aparecen en la oración y al número de nodos de los arboles-tipo que representaran la LC podados por los rasgos de las LR (subárboles-tipo). Cuanto mayor sea el tamaño de la LR y menos específicos sean los rasgos de ésta, mayor será el tiempo consumido en la comprobación.

En la oración 9 se realizan dos comprobaciones, ya que en la segunda comprobación el proceso falla:

1. <maria> SJ [humano,hembra] = [humano] Rtcc <podar> OK
2. <tapa> OD [utensilio,artificial] = [planta] Rtcc VERBAL FALLO

La oración 7 no falla y se deben realizar cinco comprobaciones:

1.<hombre> SJ [humano,macho]=[viviente] Rtcc <esperar> OK
2.<hombre> SJ [humano,macho]=[humano] Rtcc <amar> OK

3.<mujer> OD[humano,hembra]=[humano] Rtcc VERBAL OK
4.<mujer> SJ [humano,hembra]=[humano] Rtcc <creer> OK
5.<el> SUP[humano,macho]=[entidad] Rtcc VERBAL OK

y para cada una de ellas se generarán las siguientes listas de arboles-tipo de las LC podados por los rasgos de las LR:

1. [viviente([animal([humano,macho])])]
2. [humano]
3. [humano]
4. [humano]
5. [entidad ([concreta ([solido,viviente ([animal ([humano,macho])])])])]

5. CONCLUSIONES.

Del estudio comparativo anteriormente expuesto, se deduce que el método de las condiciones de emparejamiento invierte un tiempo menor en verificar las restricciones semánticas, que el método de arboles-tipo. Así pues, es más lento construir las estructuras de los subárboles y utilizar la unificación como mecanismo de validación ,que verificar el emparejamiento mediante una serie de axiomas que reflejan las condiciones que se deben cumplir.

6. BIBLIOGRAFIA.

[1] Aït-Kaci,H.y Nasr,R. (1986) LOGIN: A Logic Programming Language with Built-in Inheritance. The Journal Logic Programming

[2] Akama,S.y Isikawa,A. (1989). Semantically Constrained Parsing and Logic Programing en Meta-programming in Logic Programming. Abramson,H.y Rogers,M.H. (eds)

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

[4] Banett,J.y otros. (1990) Knowledge and Natural Language Processing. Communications of the ACM, vol.33,n.8.

[5] Beringer,H,y Porcher,F. (1989) A Relevant Schema for Prolog Extensions:CLP (Conceptual Theory). Proc.6 Internacional Conference on Logic Programming. MIT Press. (Levi,G;Martelli,M.eds)

[6] Chouraqui,E.y Dugerdil,P. (1988). Conflict solving in a frame-like multiple inheritance system. ECAI'88.

[7] Dahlgren,K y McDowell,J. (1989). Knowledge Representation for Commonsense Reasoning with Text. Computational Linguistics, vol.15, n.3.

[8] Gazdar,G.y Mellish,C. (1989). Natural Language Proccesing in PROLOG. Addison-Wesley.

[9] Grosz,J., Appelt,D., Martin,P.y Pereira,F. (1987). TEAM: An Experiment in the Desing of Transportable Natural-Language Interfaces. Artificial Intelligence, n. 32, pags:173-243.

[10] Kniht,K. (1989). Unificacion: A Multidisciplinary Survey. ACM Computing Surveys, vol.21,n. 1.

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

[12] Shimon,E. (1985). Graph Algorithms. Computer Science Press.

[13] Skvarcius,y Robinson. (1986). Discrete Mathematics with Computer Science Applications. Benjamin Cummings series in Computer Science.

[14] Ross,K.A.y Wright,C.R.B. (1988). Discrete Mathematics. Prentice-Hall.

[15] Sterling,L.y Shapiro,E. (1987). The Art of Prolog: Advanced Programming Techniques. MIT Press.

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

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