INCLUSIÓN DE CONJUNTOS BORROSOS EN EL NÚCLEO DEL SISTEMA BOUSI∼PROLOG. *

Pascual Julián-Iranzo[1] **Clemente Rubio-Manzano**[1] **Juan Gallardo-Casero**[2]

[1] Paseo de la Universidad, 4; 13071 Ciudad Real, {Pascual.Julian,Clemente.Rubio}@uclm.es
[2] Paseo de la Universidad, 4; 13071 Ciudad Real, Juan.Gallardo@alu.uclm.es

Resumen

Bousi∼Prolog es un lenguaje de programación lógica borrosa cuyo objetivo primordial es flexibilizar el proceso de respuesta a preguntas. Su mecanismo operacional es una extensión del principio de resolución SLD, que denominamos *resolución SLD débil*, donde la unificación clásica se ha sustituido por un algoritmo de unificación borrosa. En el presente trabajo se especifica un método genérico para la unificación de conjuntos borrosos que también es aplicable a otros lenguajes que se apoyen en un mecanismo de resolución débil. La idea es compilar la información relativa a los subconjuntos borrosos generando una relación borrosa entre sus etiquetas lingüísticas asociadas. Posteriormente, esta relación borrosa se emplea de forma estándar y completamente integrada en el mecanismo de unificación borrosa del sistema Bousi∼Prolog, lo que permite tratar las etiquetas lingüísticas en pie de igualdad con otras constantes definidas en el código fuente de un programa. Esta aproximación es novedosa ya que es la primera vez que se introducen los conjuntos borrosos en el núcleo de un sistema Prolog mediante el empleo de relaciones borrosas. Además, tiene la ventaja de su sencillez, ya que la inclusión se hace de manera muy natural, sin afectar a la semántica operacional del lenguaje Bousi∼Prolog y con apenas modificaciones sintácticas. Esto la convierte en una buena alternativa a la empleada por otros sistemas Prolog borrosos.

Palabras Clave: Programación Lógica Borrosa, Prolog Borroso, Unificación Borrosa, Relaciones de Proximidad, Conjuntos Borrosos.

*This work has been partially supported by FEDER and the Spanish Science and Innovation Ministry under grant TIN 2007-65749 and by the Castilla-La Mancha Regional Administration under grant PII1I09-0117-4481.

1 INTRODUCCIÓN

La Programación Lógica Borrosa es un área de investigación que estudia como introducir conceptos procedentes de la lógica borrosa en el seno de la programación lógica, con el fin de poder tratar, de manera explícita, la incertidumbre y/o la vaguedad presente en el mundo real. Existen multiples líneas de trabajo que oscilan entre las que modifican el mecanismo de resolución clásico y lo sustituyen por un mecanismo de resolución borrosa [2, 13, 17] y aquéllas que extiende el principio de resolución SLD, modificando su algoritmo de unificación clásico y sustituyéndolo por un algoritmo de unificación borrosa [1, 6, 15, 16].

Bousi∼Prolog (BPL, para abreviar) [8, 9, 10] es una extensión del lenguaje Prolog. Su semántica operacional es una adaptación de principio de resolución donde la unificación clásica se reemplaza por un algoritmo de unificación borrosa basado en relaciones de proximidad, lo que nos permite generalizar propuestas como las que aparecen en [6, 15]. Informalmente este algoritmo de *unificación débil* establece que dos términos $f(t_1, ..., t_n)$ y $g(s_1, ..., s_n)$ unifican débilmente si f y g son próximos y cada uno de sus argumentos unifican débilmente. Actualmente existen dos implementaciones[1]: una implementación de alto nivel [8], que es un meta–intérprete escrito en Prolog, y otra de bajo nivel [9, 11], que incorpora el algoritmo de unificación borrosa en una adaptación de la Máquina Abstracta de Warren y está escrita en Java. El siguiente ejemplo sirve para ilustrar la sintaxis y semántica del lenguaje.

Ejemplo 1 *Supongamos una base de datos que almacena información sobre películas y preferencias de los usuarios codificadas mediante* ecuaciones de proximidad*:*

```
%%DIRECTIVAS
:-transitivity(yes).
```

[1] Bousi∼Prolog esta disponible libremente en la URL `http://www.inf-cr.uclm.es/www/pjulian/bousi.html`

```
%%HECHOS
pelicula(el_senor_de_los_anillos,aventuras).
pelicula(terminator,accion).
pelicula(stargate,ciencia_ficcion).

%%ECUACIONES DE PROXIMIDAD
aventuras~accion=0.9.
aventuras~ciencia_ficcion=0.8.
```

Ante la pregunta "?-pelicula(stargate,accion)*" un sistema Prolog clásico fallaría. Sin embargo* Bousi∼Prolog *permite obtener la respuesta "*Yes with 0.8*", ya que este objetivo unifica débilmente con el hecho "*pelicula(stargate,ciencia_ficcion)*". Esto es posible debido a que la directiva "*transitivity*" activa, en tiempo de compilación, la generación automática de la clausura transitiva de la relación inicial, produciendo la entrada "*`accion~ciencia_ficcion = 0.8`".

El presente trabajo surge ante la necesidad de mejorar las facilidades de que actualmente dispone el sistema BPL para el tratamiento de la vaguedad y el razonamiento aproximado. Se pretende implementar el dominio conjunto borroso en el seno del sistema BPL. Dotar a Bousi∼Prolog con esta característica amplia su área de aplicación permitiendo abordar aplicaciones de control borroso, razonamiento aproximado, y en definitiva cualquier aplicación donde se necesite gestionar la vaguedad aprovechando la potencia conjunta de los conjuntos borrosos y los lenguajes declarativos. Este es pues un paso más en el objetivo de afianzar a Bousi∼Prolog como un auténtico lenguaje de programación lógica borrosa útil para el *Soft Computing*.

Para llevar a cabo este cometido se parte del concepto formal de variable lingüística [18] y se desarrolla lo que puede verse como un nuevo método para la unificación de conjuntos borrosos. Una *variable lingüística* es, en esencia, una estructura compuesta por un conjunto de términos o etiquetas lingüísticas y cuyo componente semántico son los subconjuntos borrosos asociados a las mismas. El método desarrollado consiste en convertir, en tiempo de compilación, la información almacenada por los subconjuntos borrosos en una relación borrosa sobre el conjunto de etiquetas lingüísticas. De ese modo, las etiquetas lingüísticas pasan a ser tratadas de forma análoga al resto de los símbolos del lenguaje por el mecanismo de unificación débil (que se apoya en el manejo de relaciones borrosas). Esta aproximación es útil para aquellos lenguajes de programación lógica borrosa, como LIKELOG [5], o SiLog [12], que también usan un mecanismo operacional basado en la resolución SLD débil. En este trabajo se ha estudiado con detalle como incluir esta aproximación genérica en el núcleo de la implementación de bajo nivel [9] del sistema BPL.

2 UNA APROXIMACIÓN GENÉRICA.

En este apartado proponemos una aproximación genérica para la inclusión de conjuntos borrosos en el marco de los lenguajes de programación lógica borrosa que emplean un mecanismo operacional basado en algún tipo de resolución débil, como es el caso del propio lenguaje Bousi∼Prolog. Formalmente, nuestra aproximación se apoya en los concepto de variable lingüística [18] y relación borrosa. La idea principal de nuestro planteamiento reside en manejar las variables lingüísticas mediante relaciones borrosas, esto es, transformar la información representada por una variable lingüística para que pueda tratarla una relación borrosa. Nuestra propuesta no sólo nos permitirá la inclusión de los conjuntos borrosos de manera sencilla, sino que además tiene en cuenta la distinción entre el conocimiento específico y general planteado en [14] y soluciona problemas de implementación planteados en [16].

Una *variable lingüística* es una estructura formada por la tupla $\langle X, T(X), U, G, M\rangle$, donde: X es el nombre de la variable, $T(X)$ es el conjunto de nombres de valores lingüísticos que puede tomar X, U el dominio o universo de discurso, G es una grámatica que permite generar $T(X)$ y M es una regla semántica que asocia a cada nombre de valor lingüístico x de $T(X)$ su significado, esto es, un conjunto borroso sobre U (caracterizado por su función de pertenencia μ_x).

Respecto al concepto de variable lingüística conviene hacer las siguientes puntualizaciones: $T(X)$ constituye el componente sintáctico de X, sus elementos son *términos lingüísticos*; los términos se generan mediante la gramática G (que por el momento dejamos sin especificar); se distingue entre *términos atómicos* (o *primarios*) y *términos compuesto*, que contienen términos atómicos; el dominio U es un conjunto ordinario, no necesariamente numérico; la regla semántica M permite asociar significado a los términos compuesto, mediante cómputos precisos, a partir del significado de sus términos atómicos; el significado de los términos atómicos se establece de manera axiomática asignando a cada término atómico un conjunto borroso sobre U. En otras palabras, los subconjuntos borrosos que M aplica a los términos compuesto son calculados, mientras que los que aplica a los términos atómicos son definidos (de manera subjetiva y dependiente del contexto).

Una relación borrosa binaria, $\mathcal{R}$, sobre un conjunto ordinario U, es un subconjunto borroso sobre $U \times U$ (esto es, una aplicación $U \times U \longrightarrow [0,1]$). En este trabajo estamos interesados en definir relaciones borrosas sobre el conjunto de términos, $T(X)$, asociado

a una variable lingüística, X, de forma que ésta (incluido su componente semántico) pueda tratarse a un nivel puramente sintáctico. Con este fin, procedemos del siguiente modo:

> Supongamos que $T(X) = \{x_i \mid i \in I\}$, donde I es un conjunto de índices. Para cada x_i y x_j, con $i, j \in I$, generamos la entrada de una relación borrosa sobre $T(X)$: $\mathcal{R}(x_i, x_j) = \alpha$. El grado de relación α se halla calculando la relación existente entre los subconjuntos borrosos $M(x_i)$ y $M(x_j)$ asociados a estos términos como significado. Para ello, pueden aplicarse técnicas de emparejamiento (*matching*) borroso [3, 4], empleadas con éxito en el sistema **FuzzyClips** [7].

Una vez generada la relación borrosa $\mathcal{R}$, el mecanismo operacional del lenguaje manipula la variable lingüística X y, en particular, los términos de $T(X)$ de manera totalmente estándar. Esto es, como términos (constantes) de un lenguaje de primer orden que son susceptibles de participar en un proceso de unificación débil en pie de igualdad con el resto de los otros símbolos del alfabeto de ese lenguaje.

Es importante destacar que la relación borrosa construida debe cumplir la propiedad reflexiva pero las propiedades simétrica y transitiva no tienen por que cumplirse. El empleo de este tipo de relaciones en un proceso de unificación borrosa contextual (de naturaleza diferente al procedimiento de unificación débil que nosotros empleamos) fue planteado por primera vez en [14], y ampliado en [1], donde se explica que se debe hacer distinción entre el conocimiento general y específico; por ejemplo, dada la variable lingüística `Edad` y dos de sus términos, `joven` y `entre_19_22`, una persona que sea joven no tiene por que estar entre los 19 y los 22 años, pero una persona que esté entre los 19 y los 22 años si que es una persona joven, por lo tanto, la relación entre `joven` y `entre_19_22` es distinta que la relación entre `entre_19_22` y `joven`, esto es, la relación entre estos términos, no tiene por que cumplir la propiedad simétrica.

3 CONJUNTOS BORROSOS EN EL LENGUAJE BOUSI~PROLOG.

En este apartado nos ocupamos de los detalles concretos de implementación de la estructura variable lingüística en el lenguaje Bousi~Prolog. Comenzamos con los aspectos sintácticos para pasar, después, a explicar las distintas fases de su implementación y las estructuras internas que se generan y facilitan su ejecución.

3.1 Sintaxis.

Todo lenguaje requiere introducir instrucciones específicas que permitan declarar y definir sus estructuras de datos. En el lenguaje Bousi~Prolog se hace uso de dos directivas para declarar y definir la estructura de una variable lingüística X. En realidad solamente se define el componente semántico de la misma: el dominio de discurso U y los conjuntos borrosos que se asocian a los términos primarios de $T(X)$ (el resto, como se verá, se calcula automáticamente). Por otra parte, no haremos distinción entre el componente sintáctico y semántico a efectos de notación, designando con el mismo nombre tanto a la variable lingüística como al dominio de discurso y de igual forma procederemos con los términos y sus valores. Nos remitiremos al contexto para eliminar las ambigüedades que puedan surgir.

La directiva **domain** nos permite definir el dominio o universo de discurso asociado a una variable lingüística. En general utilizamos el mismo nombre tanto para la variable lingüística como para el dominio. Por razones prácticas y porque no es una limitación severa, en esta implementación, solamente tratamos con dominios que son subintervalos de la recta real. La sintaxis concreta de esta directiva es:

```
:-domain(Domain_name(n,m,Magnitude)).
```

donde, `Domain_name` es el nombre del dominio, n y m (con $n < m$) son los límites inferior y superior del subintervalo de los números reales, $[n, m]$, y `Magnitude` es el nombre de la magnitud en la que se miden los elementos del dominio.

Ejemplo 2 *La siguiente directiva define un dominio de nombre* `age` *(edad), cuyos valores son los números comprendidos entre* 0 *y* 100*, que se miden en* `years` *(años): "*`:-domain(age(0,100,years))`*".*

La directiva **fuzzy_set** nos permite definir una lista de subconjuntos borrosos (asociados a términos primarios de una variable lingüística) sobre un domino determinado (definido previamente). La sintaxis concreta de esta directiva es:

```
:-fuzzy_set(Domain_name,[Subc1(a1,b1,c1[,d1]),
                         ...,
                         SubcN(an,bn,cn[,dn]])).
```

Los subconjuntos borrosos se definen, dando su nombre y el tipo de función de pertenencia. En estos momentos se pueden definir dos tipos de funciones de pertenencia: la *función trapezoidal*, si usamos cuatro argumentos para definir el subconjunto borroso y *la función triangular* si usamos tres argumentos. Es importante destacar que, en general, la función trapezoidal o triangular se adapta bastante bien a la definición de cualquier concepto, con la ventaja de su fácil representación y simplicidad de cálculos.

Ejemplo 3 *La siguiente directiva define una lista de subconjuntos borrosos (asociados a términos primarios de una variable lingüística* `Edad`*) cuyo universo del discurso es el dominio* `edad`*, definido anteriormente en el Ejemplo 2.*

```
:-fuzzy_set(age, [young(0,0,20,30),
                  middle(20,40,55,70),
                  old(60,80,100,100)]).
```

En concreto define tres subconjuntos borrosos asociados a los términos primarios `young` *(joven),* `middle` *(maduro) y* `old` *(viejo), cuya función de pertenencia es una función trapezoidal caracterizada por sus argumentos.*

Una vez definido un dominio y los conjuntos borrosos asociados a términos primarios, a través de la siguiente gramática se pueden generar los términos lingüísticos compuestos:

```
<Termino> ::= <Term_atomico>
              | <Term_compuesto>.
<Term_compuesto> ::= <ModifT>#<Term_atomico>.
<ModifT> ::= very | somewhat | more_or_less
             | extremely.
```

Ejemplo 4 *Para la variable lingüística* `Edad` *definida en los ejemplos 2 y 3, algunos de los términos compuestos generados a partir de los primarios serían:* `very#young`*,* `more_or_less#middle` *o* `extremely#old`*.*

Adicionalmente, en un programa BPL, pueden referenciarse lo que denominamos puntos de dominio y rangos de dominio. Un *punto dominio* es nuestra forma práctica de difuminar valores concretos del dominio de discurso[2], de forma que puedan ponerse en comparación con el resto de términos lingüísticos. Los puntos de dominio se generan mediante la siguiente gramática:

```
<Punto_dominio> ::= <Nom_dominio>#<Val_dominio>
                    | <P_D_compuesto>.
<P_D_compuesto> ::= <ModifP>#<Punto_dominio>.
<ModifP> ::= about.
```

donde `<Val_dominio>` es el conjunto de valores del dominio. Nótese que un punto de dominio puede convertirse en la etiqueta lingüística asociada a un subconjunto borroso cuando le precede el modificador `about`.

Un rango de dominio es una etiqueta para un intervalo de elementos de un dominio, comprendidos entre dos valores que lo limitan superior e inferiormente. Las etiquetas de rango de dominio se genera mediante la siguiente gramática:

```
<Rango_dominio> ::=  <Ran_dom_compuesto>
    | <Nom_dominio>#<Val_dominio>#<Val_dominio>.
<Ran_dom_compuesto> ::= <ModifR>#<Rango_dominio>.
<ModifR> ::= about.
```

Como en el caso anterior, un rango de dominio puede difuminarse mediante la adición del modificador `about`.

Ejemplo 5 *Siguiendo con el Ejemplo 2 y la variable lingüística* `Edad`*, ejemplos de puntos de dominio serían:* `age#22` *y* `about#age#22`*. Ejemplos de rangos de dominio serían:* `age#20#30` *y* `about#age#20#30`*.*

3.2 Compilación de Conjuntos Borrosos.

Una de las características más notables del sistema BPL es su capacidad de compilar toda la información relativa a las variables lingüísticas definidas por un programa. En este apartado se detallan las diferentes fases de la compilación de conjuntos borrosos.

Análisis sintáctico.

Durante la fase de análisis sintáctico se comprueba si existe algún error sintáctico en el programa y se crea un árbol sintáctico, que es la base para la generación de código posterior. Adicionalmente, en esta fase, se leen las directivas `domain` y `fuzzy_set` y se crean los dominios y los subconjuntos borrosos asociados a los mismos. La directiva `domain` dispara un procedimiento que crea un objeto de tipo dominio. Un objeto dominio se compone de un nombre, un rango y una magnitud. La directiva `fuzzy_set` lanza un procedimiento que crea objeto de tipo conjunto borroso. Un objeto de tipo conjunto borroso se compone de un dominio (referencia al dominio creado previamente) y de una lista de subconjunto borrosos. Cada subconjunto borroso se compone a su vez de una etiqueta lingüística que lo identifica (y que podrá ser usado posteriormente como una constante) y de una función de pertenencia que queda determinada por los parámetros que se pasan en los argumentos de una función trapezoidal $f(x, a, b, c, d) = max(min((x-a)/(b-a), 1, (d-x)/(d-c)), 0)$ (o triangular $f(x, a, b, c) = max(min((x-a)/(b-a), (c-x)/(c-b)), 0)$). Conceptualmente, puede considerarse que este proceso desencadena la creación de una tabla de términos lingüísticos, junto con sus significados, como la que ilustra la Figura 1.

También durante la fase de análisis sintáctico se detecta la aparición de términos lingüísticos compuestos, rangos de dominio y puntos de dominio, que se incluyen en la tabla de términos lingüísticos. El significado asociado a los términos lingüísticos compuestos es una función de pertenencia dependiente del modificador empleado sobre un determinado término primario y se resume en la tabla de la

[2] Avanzamos que, en esta tarea, se utilizarán las funciones de pertenencia definidas para cada subconjunto difuso de un dominio.

Término	Dominio	Función de pertenencia			
$TermLi_1$	Dom	a_1	b_1	c_1	d_1
$TermLi_2$	Dom	a_1	b_1	c_1	d_1
$\vdots$					
$TermLi_n$	Dom	a_n	b_n	c_n	d_n

Figura 1: Tabla de términos lingüísticos: representación en memoria de un fragmento asociado a la variable Dom.

Modificador	Desc.	Modificador	Desc.
very(y)	y^2	more_or_less(y)	$\sqrt{y}$
somewhat(y)	$y^{0.333}$	extremely(y)	y^3

Figura 2: Significado de términos lingüísticos compuestos.

Figura 2. El significado asociado a un rango de dominio `domain_name#a#b` es una función de pertenencia μ definida por una función trapezoidal $f(x, a, a, b, b)$. Esto es, $\mu(x) = 1$, si $a \leq x \leq b$, y $\mu(x) = 0$ en otro caso. En otras palabras un rango de dominio es un subconjunto *crisp*. Como se ha indicado en el Apartado 3.1, un rango de dominio admite el modificador `about`. A la etiqueta `about#domain_name#a#b` se le asocia un subconjunto borroso definido por la función trapezoidal $f(x, max(a - F, n), a, b, min(b + F, m))$, donde $F = (m - n) \times 2.5/100$ es un factor de difuminación del subconjunto crisp del que se parte. Recordamos que n y m son los extremos del intervalo $[n, m]$ que constituyen los valores del dominio de discurso. A un punto de dominio no se le asigna significado, ya que no es propiamente un subconjunto borroso (lo que simbolizaremos con una entrada nula, $\perp$, en el campo de función de pertenencia de la tabla de términos ling"uísticos). Esto nos obligará a un tratamiento particularizado de este tipo de términos en la siguiente fase. Un punto de dominio también admite el modificador `about`. A la etiqueta `about#domain_name#a` se le asocia un subconjunto borroso definido por la función triangular $f(x, max(a - F, n), a, min(a + F, m))$, donde F (definido como en el caso anterior) es un factor de difuminación que convierte el valor a en un subconjunto borroso.

Para finalizar este subapartado, hacemos notar que solamente se almacena información de aquellos términos lingüísticos que aparecen en el código fuente de un programa BPL. Esto es, no es necesario generar, por ejemplo, todos los términos compuestos derivados de uno atómico.

Ejemplo 6 *Siguiendo con el Ejemplo 2 sobre la variable lingüística* `Edad`*, Supongamos que tenemos un fragmento de una base de datos que almacena el nombre de las personas y sus edades:* `person(john,young), person(mary,age#28), person(paul,about#age#25), person(warren,very#young)` *. El resultado de esta fase de la compilación se ilustra en la tabla de la Figura 3.*

Término	Dominio	Función de pertenencia
young	*age*	μ_{young}
middle	*age*	μ_{middle}
old	*age*	μ_{old}
28	*age*	$\perp$
very_young	*age*	μ_{very_young}

Figura 3: Tabla generada tras el anlisis sintctico.

Generación de relaciones borrosas.

Una vez creada la tabla de términos lingüísticos, la siguiente fase se centra en la generación de relaciones borrosas entre términos lingüísticos con el fin de compilar toda la información semántica asociada a dichos términos. El proceso de generación se implementa mediante el siguiente algoritmo en el que, por simplicidad, nos ceñimos a los términos lingüísticos de una variable lingüística:

Algoritmo 1

Input: El conjunto de términos/significados de un dominio, $\{\langle x_i, \mu_{x_i}\rangle \mid i \in I\}$, donde I es un conjunto de índices.

Output: Un conjunto $\mathcal{R}$ de entradas que definen una relación borrosa entre términos lingüísticos.

Initialization: $\mathcal{R} := \emptyset$

For each $\langle x_i, \mu_{x_i}\rangle$ **&** $\langle x_j, \mu_{x_j}\rangle$, **with** $i, j \in I$, **do**

Case of

- $\mu_{x_i} \neq \perp$ **&** $\mu_{x_j} \neq \perp$:
  $\mathcal{R} := \mathcal{R} \cup \{\mathcal{R}(x_i, x_j) = matching(\mu_{x_i}, \mu_{x_j})\}$.
- $\mu_{x_i} \neq \perp$ **&** $\mu_{x_j} = \perp$:
  Let $x_j = domain\#u_j$ **in**
  $\mathcal{R} := \mathcal{R} \cup \{\mathcal{R}(x_i, x_j) = \mu_{x_i}(u_j)\}$.
- $\mu_{x_i} = \perp$ **&** $\mu_{x_j} \neq \perp$:
  Let $x_i = domain\#u_i$ **in**
  $\mathcal{R} := \mathcal{R} \cup \{\mathcal{R}(x_i, x_j) = \mu_{x_j}(u_i)\}$

Return $\mathcal{R}$

En el algoritmo anterior, la función de emparejamiento, *matching*, calcula el grado de relación entre dos subconjuntos borrosos $\mathcal{F}$ y $\mathcal{F}'$, en términos de una medida de su similitud S, que, siguiendo a [3, 4], se apoya en las nociones de posibilidad P y necesidad N. Más precisamente, esta función ha sido definida como sigue: $matching(\mathcal{F}, \mathcal{F}') =$

$$\begin{cases} P(\mathcal{F}, \mathcal{F}') & \texttt{si } N(\mathcal{F}, \mathcal{F}') > 0.5; \\ (N(\mathcal{F}, \mathcal{F}') + 0.5) * P(\mathcal{F}, \mathcal{F}') & \texttt{en otro caso.} \end{cases}$$

Term_1	Term_2	Grado de relación
young	*young*	1.0
young	*very#young*	0.9
young	*age#28*	0.2
young	*about#age#25*	0.25

Figura 4: Entradas para los términos del Ejemplo 8.

donde, $P(\mathcal{F}, \mathcal{F}') = max\{min(\mu_{\mathcal{F}}(u), \mu_{\mathcal{F}'}(u)) \mid u \in U\}$ y $N(\mathcal{F}, \mathcal{F}') = 1 - P(\overline{\mathcal{F}}, \mathcal{F}')$, siendo $\overline{\mathcal{F}}$ el complemento de $\mathcal{F}$ descrito por la función de pertencia: $\mu_{\overline{\mathcal{F}}}(u) = 1 - \mu_{\mathcal{F}}(u)$, $\forall u \in U$.

Ejemplo 7 *A partir de la tabla de la Figura 3, el Algoritmo 1 genera las entradas que definen una relación borrosa, $\mathcal{R}$, que se muestran en la tabla de la Figura 4.*

Concluida la fase de compilación, toda la información que contienen los conjuntos borrosos se ha almacenado en las relaciones borrosas definidas sobre el conjunto de sus etiquetas lingüísticas. Gracias a este artificio, la ejecución de un programa transcurre de forma estándar siguiendo el mecanismo de resolución débil.

Ejemplo 8 *Siguiendo con el Ejemplo 2 sobre la variable lingüística* `Edad` *y tomando los predicados del Ejemplo 6, ante la pregunta: "*`?-person(X,young).`*", el sistema BPL respondería: "*`X = john with 1.0; X = mary with 0.2; X = paul with 0.25; X = warren with 0.9`*".*

4 CONCLUSIONES

Bousi∼Prolog es un lenguaje de programación lógica borrosa cuyo objetivo es el tratamiento de la información imprecisa o vaga mediante técnicas declarativas. Todo sistema Prolog Borroso debería permitir la manipulación de conjuntos borrosos y, además, lo debería hacer de una manera sencilla y eficiente. En este trabajo se ha explicado con detalle como incluir los conjuntos borrosos en el sistema BPL. La idea general ha consistido en definir relaciones borrosas sobre el conjunto de términos, $T(X)$, asociado a una variable lingüística, X, de forma que ésta (incluido su componente semántico i.e., los subconjuntos borrosos) pueda tratarse a un nivel puramente sintáctico en el contexto de un programa lógico. Esta forma de implementación es sencilla, elegante y eficiente, permitiendo el uso de los conjuntos borrosos de manera muy natural (en el contexto de los lenguajes cuya semántica operacional es la resolución SLD débil). Por otra parte, esta aproximación es novedosa ya que es la primera vez que se introducen los conjuntos borrosos en el núcleo de un sistema Prolog mediante el uso de relaciones borrosas.

Referencias

[1] T. Alsinet and L. Godo. Fuzzy Unification Degree. In *Proc. 2nd Int Workshop on Logic Programming and Soft Computing'98, Manchester (UK)*, page 18 (1998).

[2] J. F. Baldwin, T. P. Martin, and B. W. Pilsworth. The implementation of FProlog – a fuzzy prolog interpreter. Fuzzy Sets and Systems, 23, pp 119–129 (1987).

[3] M. Cayrol, H. Farency, and H. Prade. Fuzzy pattern matching. Kybernetes, 11:103-106 (1982).

[4] D. Dubois, H. Prade, and C. Testemale. Weighted fuzzy pattern matching. Fuzzy Sets and Systems, 28:313-331 (1988).

[5] F.A Fontana and F. Formato. Likelog: A logic programming language for flexible data retrieval. In *Procs of SAC'99*, pages 260–267, (1999).

[6] F. Formato, G. Gerla, and M.I. Sessa. Similarity-based unification. *Fundam. Inform.*, 41(4):393–414 (2000).

[7] R.A. Orchard. FuzzyClips Version 6.04A. User's Guide Integrated Reasoning. Institute for Information Technology. Canada (1998).

[8] P. Julián, C. Rubio and J. Gallardo. Bousi∼prolog: a prolog extension language for flexible query answering. In: ENTCS, vol 248, pp. 131-147. Elsevier, Amsterdam (2009).

[9] P. Julián and C. Rubio. A similarity-based WAM for Bousi∼Prolog. In: LNCS, vol 5517, pp. 245–252. Springer, Heidelberg (2009).

[10] P. Julián and C. Rubio. A Declarative Semantics for Bousi∼Prolog. In: *Proc. of 11th ACM SIGPLAN Symposium on PPDP'09*. pp. 149–160 (2009).

[11] P. Julián and C. Rubio. UNICORN: A Programming Environment for Bousi∼Prolog. In: *Proc. of PROLE'09*. (2009).

[12] V. Loia, S. Senatore, and M.I. Sessa. Similarity-based SLD resolution and its implementation in an extended prolog system. In *FUZZ-IEEE*, pages 650–653, 2001.

[13] M. Mukaidono, Z.L. Shen and L. Ding. Fundamentals of fuzzy Prolog. Intl. J Approximate Reasons, 3, pp. 1080–1086 (1989).

[14] L.G. Rios-Filho and S.A. Sandri. Contextual Fuzzy Unification. In: *Proc. of IFSA'95*,pp. 81–84 (1995).

[15] M.I. Sessa. Approximate reasoning by similarity-based SLD resolution. *Fuzzy Sets and Systems*, 275:389–426 (2002).

[16] H.E. Virtanen. Linguistic Logic Programming. In T.P. Martin and F.A. Fontana (Eds): Logic Programming and Soft Computing, pp. 91–110, Wiley (1998).

[17] P. Vojtáš. Fuzzy Logic Programming. Fuzzy Sets and Systems, 124 (1): 361-370 (2001).

[18] L. A. Zadeh. The Concept of a Linguistic Variable and its Applications to Approximate Reasoning I, II and III. J. of Information Sciences 8 and 9, Elsevier (1975).