REPRESENTACION LOGICA DE BASES DE DATOS DIFUSAS: FUNDAMENTOS TEORICOS E IMPLEMENTACION

MEMORIA QUE PRESENTA
OLGA PONS CAPOTE

1994

DIRECTOR
MARIA AMPARO VILA MIRANDA

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
E INTELIGENCIA ARTIFICIAL

E.T.S. DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE GRANADA

Representación Lógica de Bases de Datos Difusas. Fundamentos Teóricos e Implementación

Olga Pons Capote

Contenidos

Introducción General

Tradicionalmente, las bases de datos han sido las herramientas computacionales más utilizadas en todos los ámbitos, para llevar a cabo las tareas de almacenamiento y manipulación de grandes cantidades de datos. Con el objeto de realizar dichas tareas eficientemente, los sistemas de bases de datos cuentan con multitud de técnicas de almacenamiento, indexación, búsqueda, etc...

A lo largo de los últimos años, han sido muchas y muy diferentes las aproximaciones que han surgido para cubrir todos estos objetivos, que se diferencian, fundamentalmente, en la forma y el tipo de almacenamiento de los datos que son capaces de gestionar.

Con respecto a la organización de los datos, los enfoques más extendidos son el Enfoque en Red, el Enfoque Jerárquico y el Enfoque Relacional (una descripción general de los tres tipos mencionados puede encontrarse en [69], [36] y [116]). Más recientemente, han surgido otros dos modelos, que son el modelo Orientado a Objetos [68] y el modelo de Bases de Datos Lógicas [57], [58], [102]. Este último, no es esencialmente distinto al modelo relacional, pero se sustenta en la lógica de predicados de primer orden para representar y manipular los datos, flexibilizando el modelo relacional original y añadiéndole capacidades de deducción automática.

Con respecto a los tipos de datos que se pueden almacenar, los modelos mencionados (en su versión tradicional) sólo aceptan datos escalares individuales (un número, una fecha, un texto,...) y, en algunos casos, algún tipo de valor omitido (nulo, desconocido o inaplicable).

Una característica de todos los modelos mencionados hasta ahora, es que sólo son capaces de manejar y representar datos ideales, en el sentido de que suponen que

la información almacenada es exacta, correcta y está bien definida. Adicionalmente, tampoco permiten obtener o consultar información en términos imprecisos.

Sin embargo, en el mundo real existe una gran cantidad y variedad de datos cuya naturaleza no permite que sean formulados de forma precisa. Como consecuencia de ésto, algunos autores se han planteado la necesidad de construir modelos de bases de datos más generales que sean capaces de manejar información expresada en términos imprecisos e inciertos. Estos sistemas, llamados, en general, de Bases de Datos Difusas, pueden clasificarse en tres grandes grupos: Modelo Relacional Difuso, Modelo de Relaciones de Similitud y Modelo Posibilístico. Las principales características que presentan estos modelos son la posibilidad de representar y manipular información cuya semántica se encuentra más próxima a la del mundo real (permitiendo el uso, no sólo de valores escalares únicos, sino de etiquetas lingüísticas tan comunes como *alto*, *más o menos grande*, *aproximadamente 12*,...., o de rangos de posibles valores como *entre 10 y 20*), y la utilización de mecanismos adecuados para consultarla y manipularla. Todos estos modelos utilizan como fundamento la Teoría de Subconjuntos Difusos de Zadeh [126]. La mayoría de las propuestas teóricas realizadas en este sentido, se fundamentan en el modelo relacional.

Paralelamente, los sistemas de programación lógica tambi'en han ido evolucionando en ese sentido, incorporando elementos de la *lógica fuzzy* y surgiendo, lo que podemos llamar, sistemas de programación lógica difusa. En este sentido, los trabajos más relevantes son [5], [92], [107], [91], orientados, básicamente, en la definición de un *Prolog difuso*.

Partiendo de estas aproximaciones, podemos fijar ya nuestros objetivos con más propiedad.

Objetivos

Como hemos indicado en la introducción anterior, con posterioridad al modelo rela-

cional, se desarrolló un modelo lógico paralelo con el objetivo de ampliar sus posibilidades, dotándolo, principalmente, de la capacidad de deducción automática de información a partir de la que hay físicamente almacenada en la base de datos propiamente dicha. En este punto, nos planteamos ¿por qué no ampliar el modelo lógico clásico para bases de datos de manera que admita información imprecisa e incierta?. Obsérvese que, de esta forma, estamos aprovechando las ventajas de ambas aproximaciones, la difusa y la lógica, de manera que tenemos posibilidad tanto de manipular información imprecisa e incierta como de hacer deducción automática a partir de ella.

Nuestro trabajo consistirá, pues, en llevar a cabo esta tarea. Para conseguirlo, consideraremos las siguientes etapas:

- La definición de un modelo teórico lógico con capacidad para representar información imprecisa. Este modelo ha de ser lo suficientemente flexible y general como para permitir integrar las características más relevantes de cada uno de los modelos de bases de datos difusas que se encuentran en la literatura.

- Estudiar y analizar una posible extensión del modelo anterior para la manipulación coherente y correcta de cualquier tipo de omisión de datos, esto es, información disyuntiva, desconocida, inaplicable o nula.

- Dotar a dicho modelo de un sistema de deducción adecuado a los datos que se van a almacenar. Esto conlleva la posibilidad de definir y utilizar reglas de carácter impreciso que serán aplicadas sobre los datos y describir un mecanismo de deducción a partir de dichas reglas.

- Llevar a cabo la construcción de un prototipo experimental, que muestre las posibilidades de dicho modelo teórico. Previamente, han de estudiarse qué adaptaciones es necesario realizar sobre el modelo teórico para convertirlo en un modelo computacional no sólo correcto, sino eficiente.

Los capítulos principales se hallan organizados secuencialmente según estas tareas. El primer capítulo, se dedica a recopilar las características más destacadas de los modelos clásicos de bases de datos y su representación lógica, para lo cual se da, así mismo, una pequeña introducción a la lógica de predicados de primer orden. El segundo, lleva

a cabo una recopilación análoga, en la que se estudian los modelos más relevantes de bases de datos difusas que aparecen en la literatura. También se ofrece un resumen de los conceptos más interesantes, desde el punto de vista de nuestro trabajo, de la lógica difusa, sobre la que se fundamentan dichos modelos. En el capítulo tercero, se lleva a cabo la definición de un modelo lógico de bases de datos difusas, con capacidad de manipular información imprecisa y omitida. También se muestra en este capítulo como quedan englobados en dicho modelo, los modelos difusos más destacados de la literatura. En el capítulo cuarto, se da una definición y una semántica de lo que llamamos *reglas difusas*, que serán utilizadas como base para la definición de un esquema de deducción adecuado al modelo introducido, y del que se derivan, por otra parte, algunas propiedades deseables. Así mismo, se introduce la posibilidad de utilizar las reglas para la definición intensiva de atributos virtuales de la base de datos, esto es, la posibilidad de ampliar la base de datos (esquema e información) en virtud de la información almacenada. Por último, en el capítulo quinto, se analizan las características más destacadas de su implementación (que se lleva a cabo en lenguaje Prolog) y se muestra y se discute un amplio ejemplo.

Capítulo 1

Preliminares I: Conceptos Clásicos

Con este capítulo se pretende sentar las bases sobre las que se fundamenta nuestro trabajo. Para ello se ha estructurado en tres partes claramente diferenciadas:

- En la primera parte, se introducen conceptos básicos sobre la lógica de primer orden, se establece la notación a utilizar en los sucesivos capítulos y se dan unas nociones sobre el lenguaje Prolog y su entorno, justificándose su elección por nuestra parte como lenguaje fundamental en nuestra tarea de desarrollo e implementación.

- En la segunda parte, se resume brevemente el modelo relacional para bases de datos, haciendo especial hincapié en el cálculo relacional como lenguaje de consulta, por su directa relación con la lógica de primer orden y por ser el lenguaje (con algunos elementos adicionales) que adoptaremos para consultar finalmente nuestra base de datos lógica difusa. Así mismo, se introduce el modelo lógico de representación de BDR y se comentan las principales ventajas de dicho modelo respecto a la representación, manipulación y restricciones de los datos, así como su flexibilidad de cara a futuras amplicaciones.

- Por último, se considerarán las aproximaciones que aparecen en la literatura para acoplar las características del modelo relacional a un entorno de programación lógica, señalando las distintas partes de la arquitectura y las peculiaridades de cada sistema.

1.1 La Lógica de Predicados de Primer Orden y el Lenguaje Prolog

1.1.1 La Lógica de Primer Orden

◇ Introducción

La lógica es una herramienta muy importante para el análisis y la representación de sentencias, e investiga si las suposiciones hechas implican determinadas conclusiones, independientemente de su veracidad o falsedad, e independientemente del tema al que se refieran. Esto es, la lógica se basa en la representación formal de sentencias sin preocuparse más que de las relaciones existentes entre ellas.

Para demostrar que unas suposiciones implican una conclusión, se recurre a la realización de una prueba, consistente en un conjunto de pasos de inferencia. Para poder realizarlo, es requisito indispensable que el lenguaje utilizado sea simple y sin ambigüedades, lo que ha llevado a emplear un lenguaje simbólico.

Mientras que la Lógica Proposicional atiende, únicamente, a la forma de las fórmulas (los conectivos que intervienen) para llevar a cabo el proceso de inferencia, en Lógica de Predicados, algo se considera verdadero o falso teniendo en cuenta, además de los conectivos que intervienen en la fórmula considerada, la estructura interna de las proposiciones que intervienen en la misma, aunque no su significado. Así, por ejemplo, influirá si alguna de las proposiciones lleva cuantificadas sus variables y de qué forma. La lógica de predicados lleva a cabo, pues, un análisis lógico de las proposiciones. Nuestro interés se centra en la lógica de predicados, ya que es la que fundamenta los sistemas de programación lógica.

◇ El Lenguaje de la Lógica de Primer Orden

La Lógica matemática [78] [67] [37] está basada en un lenguaje objeto, una semántica o interpretación y una teoría de pruebas. Como lenguaje objeto utilizaremos uno de primer orden. De manera informal, un lenguaje de Primer Orden de la Lógica de

Predicados, es el conjunto de todas las sentencias que se pueden construir de acuerdo con un alfabeto preestablecido.

Un lenguaje de primer orden L, viene dado por un par $L = (A, W)$ donde A es un alfabeto de símbolos y W es un conjunto de fórmulas sintácticamente correctas llamadas fórmulas bien formuladas (fbf), que se representan mediante símbolos del alfabeto A. A estará compuesto por los siguientes símbolos:

- *Símbolos de puntuación*: ( ) , ; : .
- *Símbolos de constantes*: Usaremos las minúsculas primeras del alfabeto, a,b,c,...
- *Símbolos de variables*: Usaremos las minúsculas últimas del alfabeto, u,v,w,...
- *Símbolos de funciones*: Usaremos las letras f,g,h,...
- *Símbolos de predicados*: Usaremos letras o palabras en mayúsculas.
- *Símbolos de los conectivos lógicos* (también llamados constantes lógicas): $\longrightarrow$, $\wedge$, $\vee$, $\neg$.
- *Cuantificadores*: $\exists$, $\forall$.

Definición 1.1 . *Sea L un lenguaje. Definimos un* **término** *de L de la siguiente manera:*

- *Toda constante es un término.*
- *Toda variable es un término.*
- *Si f es un símbolo funcional n-ario, y $t_1, t_2, \ldots, t_n$ son términos, entonces $f(t_1, t_2, \ldots, t_n)$ es un término.*
- *Los únicos términos son los que se obtienen por las reglas anteriores.*

Definición 1.2 . *Si P es un predicado n-ario de A y $t_1, t_2, \ldots, t_n$ son términos, $P(t_1, t_2, \ldots, t_n)$ es una* **fórmula atómica**.

Definición 1.3 . *Una* **fórmula bien formulada** *se define recursivamente de la siguiente forma:*

1. *Una fórmula atómica es una fbf.*
2. *Si w_1 y w_2 son fbf, también lo son $(w_1 \wedge w_2)$, $(w_1 \vee w_2)$, $(w_1 \longrightarrow w_2)$ y $\neg(w_1)$.*
3. *Si x es una variable y w es una fbf entonces $(\exists x,\ w)$ y $(\forall x,\ w)$ son fbf.*

Definición 1.4 . *Sea $(Qx,\ W)$ una fbf, donde Q es un cuantificador. Llamaremos* **ámbito del cuantificador Q** *a la fbf W.*

Definición 1.5 . *Una fbf se dice* **cerrada** *si no contiene ninguna variable libre, es decir, sólo tiene variables cuantificadas y constantes.*

Una ocurrencia de una variable x en una fórmula se dice **ligada**, si está en el dominio de un cuantificador. Se dice **libre** si no está ligada.

Definición 1.6 . *Se denomina* **sentencia** *a una fbf en la que toda variable está ligada.*

Definición 1.7 . *Se llama* **sustitución elemental de una variable x por un término** *t, a una aplicación que a cada fórmula le hace corresponder la fórmula que se obtiene sustituyendo las ocurrencias libres de x por t.*

Tanto en el caso de la Lógica Proposicional, como en el caso de la Lógica de Predicados (que es la que nos ocupa), la derivación de una conclusión partiendo de una serie de premisas, puede realizarse desde dos puntos de vista diferentes: el *Sintáctico* y el *Semántico* [88].

◇ El Método Sintáctico o Teoría de Pruebas

La Teoría de Pruebas trata, esencialmente, sobre la *derivabilidad* de una sentencia en el contexto de un conjunto de reglas, esto es, tomando como punto de partida un conjunto

inicial de sentencias S del lenguaje L, y aplicando una serie de reglas R, se obtiene un conjunto de sentencias nuevas s. Esta relación de derivabilidad se define de la siguiente forma:

$$\vdash = \{< S, s > \mid S \subset L,\ s \in L \text{ y } s \text{ es derivable de } S \text{ usando } R\}$$

De manera informal, diremos que a las sentencias de partida se les denomina *axiomas*, a las sentencias derivadas *teoremas* y, a las reglas utilizadas, *reglas de inferencia*. De estas últimas, las más conocidas y utilizadas son el *Modus Ponens* y la *Generalización*.

Por tanto, desde el punto de vista sintáctico, una lógica de primer orden es un sistema formal formado por un lenguaje objeto, un conjunto de axiomas y una serie de reglas de inferencia

Supondremos que el lenguaje es uno fijo, de manera que el conjunto de todas las sentencias sobre L, llamémoslo $P(L)$, sea también fijo.

Aunque en el entorno de la programación lógica, los axiomas de partida son los hechos (afirmaciones) que definen el problema concreto a tratar, en el marco de la lógica de primer orden, se selecciona un conjunto inicial de *axiomas lógicos*, a partir del cual se generarán todas las sentencias válidas.

– Concepto de Demostración

Definición 1.8 . *Sea $\Gamma \subset P(L)$ un conjunto de proposiciones sobre el lenguaje L. Una* **demostración** *a partir de Γ es una sucesión finita de proposiciones $< B_1, B_2, \ldots, B_n >$ que cumple alguna de las siguientes condiciones:*

1. $B_i \in \Gamma$
2. $B_i \in A_x$
3. $\exists\ j, k < i$ *tales que* $B_j = B_k \longrightarrow B_i$

para todo i verificando $1 \leq i \leq n$.

Una demostración de w a partir de Γ es una demostración a partir de Γ que acaba en w. Para indicar que existe tal demostración, usaremos la notación $\Gamma \vdash w$.

Definición 1.9 . *Sea A una proposición. Diremos que A es un* **teorema o ley lógica** *si se verifica que $\emptyset \vdash A$, y lo notaremos simplificadamente como $\vdash A$.*

Definición 1.10 . *Sea $P(L)$ el conjunto de las posibles proposiciones sobre el lenguaje L, $\mathcal{P}(P(L))$ el conjunto de las partes de $P(L)$ y $\Gamma \subset P(L)$. Se define la* **aplicación deducción**, *$Ded : \mathcal{P}(P(L)) \longrightarrow \mathcal{P}(P(L))$, como:*

$$Ded(\Gamma) = \{w \in P(L) : \Gamma \vdash w\}$$

Propiedades:

1. $\Gamma \subset Ded(\Gamma)$
2. Si $\Delta \subset \Gamma$, entonces $Ded(\Delta) \subset Ded(\Gamma)$
3. $Ded(Ded(\Gamma)) = Ded(\Gamma)$
4. $Ded(\Gamma) = \bigcup_{\Gamma_i \subset \Gamma} Ded(\Gamma_i)$ cuando los Γ_i son finitos. Esto es, todo lo que se deduce de Γ es igual a la unión de todo lo que se deduce de cada subconjunto finito de Γ.

Definición 1.11 . *Un conjunto de proposiciones Γ se dice* **inconsistente** *si $Ded(\Gamma) = P(L)$, es decir, si cualquier proposición es una consecuencia de Γ. Si ésto no ocurre, se dice que Γ es consistente.*

Teorema 1.1 . *Las tres condiciones siguientes son equivalentes:*

1. *Γ es inconsistente.*
2. *$\forall\ w_1 \in P(L),\ \Gamma \vdash w_1 \wedge \neg w_1$.*
3. *$\exists\ w_1 \in P(L),\ \Gamma \vdash w_1 \wedge \neg w_1$.*

Definición 1.12 . *Se llaman* **tautologías** *aquellas proposiciones cuyo valor de verdad es la constante 1.*

Definición 1.13 . *Se llaman* **contradicciones** *aquellas proposiciones cuyo valor de verdad es la constante 0. Las que no son contradicciones, se denominan* **satisfacibles**.

Teorema 1.2 . *Todo teorema de $P(L)$ es una tautología.*

Teorema 1.3 (de Adecuación). *Si w es una fbf de $P(L)$ y w es una tautología, entoces w es un teorema de $P(L)$.*

◇ El Método Semántico o Teoría de Modelos

De manera informal, una interpretación consiste en especificar un conjunto no vacío del cual van a tomar sus valores tanto las constantes como las variables. En una interpretación de dominio U, una fbf cerrada es verdadera o falsa.

Denominamos **L-Estructura** a una estructura (U,K,F,P) donde:

- U es un conjunto no vacío, llamado dominio o universo de la interpretación.
- K es un subconjunto de U del que toman sus valores las constantes del lenguaje.
- F es un conjunto de funciones $f^U : U^U \longrightarrow U$, una por cada símbolo funcional del lenguaje.
- P es un conjunto de predicados R^U, uno por cada símbolo de predicado del lenguaje, del que toman sus valores las relaciones n-arias.

Para que las proposiciones abiertas puedan interpretarse, será necesario sustituir las variables libres por identificadores.

Definición 1.14 . *Sea V el conjunto de todas las variables del lenguaje. Se llama* **asignación en U**, *a una aplicación de V en U. De manera informal, es el proceso de sustituir todas las variables de una fórmula por elementos del universo de discurso, de manera que ésta adquiera significado.*

Definición 1.15 . *Una* **interpretación de un lenguaje** L *es la pareja formada por una L-Estructura en U y una asignación en U.*

Gracias a la interpretación, puede asignarse un valor verdadero (1) o falso (0) a cualquier fórmula de nuestro lenguaje.

– Interpretación de fórmulas

Partiendo de una interpretación I, vamos a asignar a cada fórmula el valor 0 ó 1, según sea falsa o verdadera, respectivamente, de la siguiente forma:

1. $I(t_1 = t_2) = \begin{cases} 1 & \text{si } I(t_1) = I(t_2) \\ 0 & \text{en otro caso} \end{cases}$
2. $I(R(t_1, t_2, \ldots, t_n)) = \begin{cases} 1 & \text{si } < I(t_1), I(t_2), \ldots, I(t_n) > \in P \\ 0 & \text{en otro caso} \end{cases}$
3. $I(\neg B) = 1 + I(B)$
4. $I(A \wedge B) = I(A) * I(B)$
5. $I(A \vee B) = I(A) + I(B) + I(A) * I(B)$
6. $I(A \longrightarrow B) = 1 + I(A) + I(A) * I(B)$
7. $I(\forall x, B) = \begin{cases} 0 & \text{si } \exists\ b \in \text{U: } I' = (\text{U}, a(x/b)) \text{ verifica que } I'(B) = 0 \\ 1 & \text{en otro caso} \end{cases}$

Nota: $a(x/b)$ es la asignación que sustituye la variable x por el valor b.

Definición 1.16 . *Un* **modelo** *de un conjunto de fbf es una interpretación en la que todas las fbf son verdad. Una fbf w es una* **consecuencia lógica** *de un conjunto de fbf W, si y sólo si w es verdad en todos los modelos de W, y lo notamos* $W \models w$. *Notaremos* $Con(\Gamma)$ *al conjunto* $\{w \in P(L) \ : \ \Gamma \models w\}$.

Una L-Estructura en U es modelo de una fórmula w, si $I(w) = 1$ *para toda interpretación I=(U,a).*

Los dos teoremas que presentamos a continuación, son dos importantes resultados que nos garantizan que los métodos sintáctico y semántico son equivalentes, esto es, que todos los teoremas son fórmulas universalemente válidas y vicerversa.

Teorema 1.4 (de Coherencia). *Si* $\Gamma \subset P(L)$, $w \in P(L)$ *y* $\Gamma \vdash w$, *entonces* $\Gamma \models w$, *esto es,* $Ded(\Gamma) \subset Con(\Gamma)$.

Teorema 1.5 (de Completitud de Gödel). *Si* $\Gamma \subset P(L)$, $w \in P(L)$ *y* $\Gamma \models w$, *entonces* $\Gamma \vdash w$, *esto es,* $Con(\Gamma) \subset Ded(\Gamma)$.

◇ Forma Clausal de la Lógica

La forma clausal de la Lógica [70] es el sub-lenguaje en el que, normalmente, se escriben todos los programas lógicos. La principal ventaja de la forma clausal estriba en que la estructura de las sentencias es muy regular y, por supuesto, que cualquier sentencia del lenguaje puede ser expresada en forma clausal sin ninguna pérdida de expresividad. Todo esto, lo veremos con detalle a lo largo de la sección.

Definición 1.17 . *Se denomina* **literal** *a una variable proposicional o su negación. Se denomina* **literal positivo** *a una variable proposicional y* **literal negativo** *a la negación de una variable proposicional.*

Definición 1.18 . *Se denomina* **cláusula** *a una disyunción de literales. Notaremos como* $Cl(L)$ *al conjunto de las cláusulas de del lenguaje* L. *Entre las cláusulas distinguiremos un conjunto de ellas denominadas* **de Horn** *[71], que se caracterizan por tener, como máximo, un literal positivo. A las que tienen un literal positivo se les denomina cláusulas de Horn* **con cabeza** *y a las que no,* **sin cabeza o descabezadas**.

La **cláusula vacía** es una cláusula sin cabeza que no contiene ningún literal. Normalmente se nota con el símbolo especial $\square$.

Teorema 1.6 (de la Forma Normal Conjuntiva). *Toda proposición es equivalente a una proposición que es conjunción de cláusulas.*

– Conversión de sentencias a forma clausal

En muchos casos, cuando tenemos que formular un problema con el objeto de utilizar un programa lógico para resolverlo, es conveniente expresar los hechos y reglas de partida directamente en forma de cláusulas. Existe un algoritmo [124] [93] muy sencillo para convertir cualquier sentencia de la lógica de primer orden en un conjunto de cláusulas. En este proceso de conversión, aseguramos, gracias a un adecuado renombrado de variables, que no hay dos cuantificadores de la sentencia original que cuantifiquen la misma variable. El algoritmo sería el siguiente:

1. En primer lugar, reescribir las fórmulas del tipo

$$w_1 \longleftrightarrow w_2 \quad como \quad (w_1 \longrightarrow w_2) \ \wedge \ (w_2 \longrightarrow w_1)$$

2. A continuación, escribir cada fórmula

$$w_1 \longrightarrow w_2 \quad como \quad \neg w_1 \vee w_2$$

3. Aplicar las leyes de la negación, sustituyendo:

$$\neg(\exists x)w \quad por \quad (\forall x)\neg w$$

$$\neg(\forall x)w \quad por \quad (\exists x)\neg w$$

$$\neg(w_1 \vee w_2) \quad por \quad \neg w_1 \wedge \neg w_2$$

$$\neg(w_1 \wedge w_2) \quad por \quad \neg w_1 \vee \neg w_2$$

$$\neg\neg w \ por \ w$$

4. Aplicar la distributividad del operador $\vee$:

$$w_1 \vee (w_2 \wedge w_3) \quad como \quad (w_1 \vee w_2) \wedge (w_1 \vee w_3)$$

$$w_1 \vee (\forall x)w_2 \quad como \quad (\forall x)(w_1 \vee w_2)$$

$$w_1 \vee (\exists x)w_2 \quad como \quad (\exists x)(w_1 \vee w_2)$$

En los dos últimos casos, hay que tener en cuenta que x no puede aparecer en w_1.

5. Aplicar la distribución del cuantificador $\forall$, reescribiendo

$$(\forall x)(w_1 \wedge w_2) \quad como \quad (\forall x)w_1 \wedge (\forall x)w_2$$

En este punto, si no quedan cuantificadores existenciales en la sentencia, el proceso está practicamente terminado y pasaremos directamente al paso 7.

6. Reemplazar todas las fórmulas (sin variables libres) de la forma

$$(\forall x_1)\ldots(\forall x_n)\ (\exists y)w(y) \quad por \quad (\forall x_1)\ldots(\forall x_n)\ w(f(x_1,\ldots,x_n))$$

donde f es cualquier símbolo de función. Este proceso se denomina **Skolemización**, y el nuevo símbolo de función introducido se llama **función de Skolem**. En el caso en el que $n = 0$, la sustitución será:

$$(\exists y)w(y) \quad por \quad w(c)$$

donde c es cualquier símbolo de constante, y recibe el nombre de **constante de Skolem**.

7. Aplicar de nuevo, si es posible, el paso 5. En este punto, la sentencia original habrá quedado transformada en una conjunción de cláusulas. Finalmente, para presentar esta fórmula como un conjunto de cláusulas cuantificadas de manera implícita, se eliminan todos los cuantificadores universales y las ocurrencias del operador $\wedge$.

Es importante notar que, siguiendo los pasos del 1 al 5 y el 7, el algoritmo preserva la equivalencia, esto es, si S es la sentencia original y C el conjunto de cláusulas resultante, se verfica que $C \equiv S$. Sin embargo, si es necesario llevar a cabo el proceso de Skolemización, esta equivalencia no se mantiene, verificándose tan solo las propiedades siguientes:

(i) C es satisfacible si y sólo si S es satisfacible.

(ii) $C \models S$.

◇ Resolución

La resolución [93] [53] es una regla de inferencia aplicable a sentencias en forma clausular que, a cada par de cláusulas, verificando una determinada condición, le hace corresponder otra cláusula.

Comenzaremos introduciendo la resolución para el caso más sencillo, el de la lógica proposicional para, posteriormente, pasar a comentar las particularidades que ésta tiene para el caso de la Lógica de Predicados.

La condición a verificar por cada par de cláusulas al que se va a aplicar la Resolución, es que debe aparecer en ambas el mismo literal, pero en una de ellas negado y en la otra no (literales complementarios). Se denomina **resolvente** a la cláusula que se obtiene, y está formada por la disyunción de todos los literales de las dos cláusulas originales, eliminado el literal común. Las dos cláusulas utilizadas en la consecución de una resolvente se denominan *padres* de dicha resolvente. Una resolvente puede ser tomada, a su vez, como padre de una nueva resolvente.

Por ejemplo, si partimos de las cláusulas $p \vee w_1$ y $\neg p \vee w_2$, donde p es una variable proposicional y w_1 y w_2 son cláusulas, aplicando la regla de resolución nos quedaría la cláusula $w_1 \vee w_2$ (los literales p y $\neg p$ serían eliminados).

Definición 1.19 . *Una demostración por resolución de una cláusula C, a partir de un conjunto Γ (hipótesis), es una sucesión finita $< B_1, B_2, \ldots, B_m >$ de cláusulas tales que $B_m = C$, y para todo i, verificando $1 \leq i \leq m$, se cumple que $B_i \in \Gamma$, o bien que B_i es la resolvente de dos cláusulas anteriores.*

Notaremos por $Res(\Gamma)$ al conjunto de cláusulas que puede obtenerse a partir de Γ mediante resolución.

Para el caso de la Lógica de Predicados, es necesario utilizar otra regla de inferencia, denominada **instanciación universal**, que tiene la siguiente forma:

$$(\forall x)\ w(x) \vdash (\forall y_1) \ldots (\forall y_m)\ w(x)(x/t)$$

para cualquier término t donde $y_1 \ldots y_m$ son las variables, si las hay, de t.

El proceso de Resolución tiene tres propiedades importantes, que se verifican, tanto en el caso de la Lógica Proposicional como en el de la Lógica de Predicados, y son las siguientes:

1. La Resolución es *consistente*: las cláusulas utilizadas como padres implican la resultante.
2. Una resolvente es la cláusula vacía ($\square$) si y sólo si sus padres son literales (cláusulas unitarias) complementarios.
3. Para cualquier conjunto de cláusulas de Horn, la cláusula vacía se obtiene mediante Resolución, sólo si el conjunto inicial es insatisfacible. En virtud de esta propiedad, se dice que el proceso de Resolución es *completo respecto a la refutación.*

◇ Algunos Aspectos sobre la Demostración Automática de Teoremas. El Problema de la Decidibilidad

Por lo general, un conjunto inicial de claúsulas que sea insatisfacible puede llevar a la cláusula vacía por distintos caminos aplicando resolución. Se han realizado muchos estudios acerca de cómo llevar a cabo este proceso de manera que resulte lo más eficiente posible. La forma de restringir el proceso de resolución que se utiliza normalmente para probar cláusulas negativas (queries) por medio de programas expresados en forma clausular se denomina **SLD-Resolución** [64]. Las siglas antepuestas tienen el siguiente significado:

- **L**ineal, indicando que cada paso del proceso de resolución utiliza como uno de los padres del siguiente paso la resolvente más reciente y, como segundo padre, alguna resolvente anterior o bien una cláusula de partida.
- **S**elección, indicando que en cada paso del proceso se aplica una *regla de computación* para elegir el literal adecuado al primer padre.
- **D**efinido, indicando que todas las cláusulas del programa son definidas (de Horn).

Los intérpretes de los lenguajes de programación lógica, suelen llevar implementados una regla de computación fija, aunque existen algunos que ofrecen al usuario la

posibilidad de elegir una de entre varias. El utilizar una regla u otra sólo variará la eficiencia del proceso, nunca su resultado. Dado un programa P y una consulta Q, la regla de computación determina un árbol de búsqueda cuya raíz es Q. La búsqueda más simple de todas se da cuando el árbol no contiene ramas, de forma que sólo existe un posible camino a seguir para dar con la solución. En este caso, se dice que el proceso es **determinista**. Por lo general no es este el caso, y habrá distintas ramas y, por tanto, distintos caminos a seguir, denominándose entonces el proceso **no determinista**. En este último caso, suele usarse alguna estrategia de búsqueda que determina la mejor forma de exploración del árbol. La estrategia más utilizada por los lenguajes de programación lógica es la búsqueda *secuencial primero en profundidad con vuelta atrás.* Esta estrategia se describe en profundidad en la sección 1.1.2 sobre lenguaje Prolog. Teniendo en cuenta todos estos conceptos y considerando que:

- un programa es un conjunto de cláusulas de Horn,
- una consulta es una cláusula negativa,
- un cálculo o computación es una derivación utilizando SLD- Resolución y
- una ejecución es una búsqueda exhaustiva sobre un árbol,

nos planteamos ahora, ¿qué clase de problemas podremos representar y resolver mediante este formalismo?, ¿sigue siendo el proceso de inferencia completo y consistente?, ¿qué clase de problemas no permite representar o resolver este formalismo?... Trataremos de responder algunas de estas preguntas a continuación. Si se desea un estudio más formal sobre el tema, ver [67] [63].

El concepto de *computabilidad* [63] [62] fue definido por Alan Turing de la siguiente forma: *"Las funciones computables son aquellas que pueden calcularse por medio de una máquina de Turing Universal"*. Posteriormente, Alonzo Church afirmó, en su conocidad tesis, que dichas funciones eran las únicas funciones calculables en cualquier tipo de máquina.

Dado que la forma clausular de Horn es capaz de expresar todos los problemas calculables (o computables) [70], podría plantearse ahora la cuestión de si es un formalismo adecuado *desde el punto de vista práctico.* Teniendo en cuenta que hay procesos en los

que se hace necesario razonar con la negación o con disyunciones, una de las restricciones más importantes que nos encontramos es que no existe un mecanismo directo de conversión de cláusulas a cláusulas de Horn, ya que se hace necesario llevar una cabo una recodificación del programa original, esto es, la creación de un nuevo programa en forma clausular de Horn que *simule* el razonamiento del primero. *Desde el punto de vista teórico*, lo que sí está demostrado es que cualquier problema computacionalmente resoluble puede resolverse aplicando resolución en forma clausal. Esto equivale a afirmar que cualquier implicación lógica que se verifique puede confirmarse por este mismo método, de la misma manera que puede verificarse la validez de una sentencia. Pero, ¿cómo responde este sistema ante un problema irresoluble, una implicación que no se verifica o una sentencia que no es válida?. La respuesta a esta pregunta es que, para la lógica de primer orden en general (no así para la lógica proposicional), el problema de la *validez* es sólo *semidecidible*, esto es, es imposible construir un algoritmo que sea capaz de distinguir sentecias válidas y no válidas. En el caso de que la entrada no sea válida, la respuesta del sistema puede ser, o un fallo (parada) o un bucle infinito.

◇ Conclusiones

En definitiva, dada la homogeneidad y flexibilidad de la lógica, es muy amplia la gama de posibilidades que ésta ofrece como herramienta de representación y manipulación de conocimiento. Entre otras, se pueden destacar las siguientes cualidades:

- Permite representar y manipular de forma natural y directa información disyuntiva.

- Gracias a la lógica puede comprobarse la correctitud de la bases de datos y asegurarse el mantenimiento de la integridad de los datos, ya que todas estas restricciones pueden especificarse como fórmulas bien formuladas del lenguaje subyacente (reglas) que deben verificarse en cualquier estado de la base de datos.

- La lógica ofrece mecanismos de deducción fácilmente automatizables y de gran eficacia que permiten obtener información adicional a la contenida explícitamente en una Base de Datos o una Base de Conocimiento.

– La lógica en sí puede ser usada como lenguaje de consulta, de representación y de manejo de los datos [72]; es por tanto, una herramienta multi-propósito.

1.1.2 El Prolog como Lenguaje de Programación Lógica

◇ Introducción

La *Programación Lógica* [39] [64] representa un punto de convergencia entre la Lógica, la Demostración Automática de Teoremas y las Ciencias de la Computación y está basada en el cálculo de predicados de primer orden, generalmente sobre el subconjunto de las cláusulas de Horn. La orientación computacional dada a las dos primeras disciplinas citadas se debió, especialmente, a Carl Hewitt, Alan Colmerauer y Robert Kowalski.

La programación lógica, está fundamentada en el Principio de Resolución (enunciado por J. Alan Robinson en 1965). A principios de los 70 Kowalski formuló la interpretación de la lógica en forma clausular desde un lenguaje de programación, lo que sirvió de punto de partida a un grupo de investigadores de la Universidad de Aix -Marsella- (entre sus miembros Colmerauer y Roussel) para implementar el primer sistema Prolog. A partir de ahí, se fueron sucediendo estudios y publicaciones sobre el tema, que dieron lugar, en 1981 al Proyecto de Sistemas de Computación de Quinta Generación (basado en Prolog), en 1984, a la aparición de la Revista sobre Programación Lógica (bajo la dirección de Alan Robinson) y, en 1986, a la inauguración de la Asociación de la Programación Lógica.

Este rápido y fructífero crecimiento, se ha debido a una serie de cualidades clave que presenta la programación lógica [19], entre ellas:

– Permite formular las restricciones y suposiciones del dominio de un problema de manera muy directa e independiente de la implementación. Además, dicha información es fácilmente recodificable.

– Permite dar una caracterización matemática precisa de las relaciones existentes entre un programa y los resultados que se derivan de él, un programa y sus especificaciones y varios programas entre sí.

- Ofrece un entorno común a la tecnología del software, ya que el mismo formalismo es utilizado para la construcción y manipulación de programas, la representación de las especificaciones, las bases de datos y otras herramientas software adicionales.
- Puede ser fácilmente modificado o extendido para representar distintas formas de conocimiento (meta-conocimiento, conocimiento de orden superior, formalismos no lógicos,...).
- Una característica esencial de las implementaciones hechas de la programación lógica es la eficiencia con que se lleva a cabo el proceso de resolución (ya que este proceso con facilidad puede dar lugar a una explosión combinatoria si no implementa de la forma adecuada). Todo ello, ha llevado a considerarla, no sólo un marco propicio para representación de conocimiento, sino también un modelo computacional para el desarrollo de Bases de Datos y Bases de Conocimiento inteligentes.

El lenguaje Prolog (su nombre procede de las palabras **PRO**gramming in **LOG**ic) es el lenguaje de programación lógica más popular y extendido, y surgió como simplificación de técnicas generales de demostración de teoremas. Desde que fue implementado por primera vez por Colmerauer en 1973, (para el procesamiento de lenguaje natural), se han ido sucediendo versiones cada vez más completas en las que se ofrecen gran cantidad de predicados procedurales de Entrada/Salida, así como facilidades para el acceso a Bases de Datos [79], definición de operaciones, interfaces amigables con el usuario, etc... En la siguiente sección pasamos a describir algunas de sus características más relevantes.

◇ Principales Características del Lenguaje Prolog

El lenguaje Prolog [22] [61] [111] [137] es una implementación del subconjunto de las cláusulas de Horn positivas. Un programa en Prolog se corresponde con un conjunto de hipótesis, mientras que una consulta sería un teorema a ser probado por el demostrador de teoremas, que en este caso emplea como herramienta la unificación.

En lo que sigue de este apartado, daremos una breve descripción de las características más relevantes del lenguaje Prolog y su entorno.

La estructura del lenguaje está basada en las nociones de *átomo*, *variable*, *functor*, *predicado*, *término*, *literal* y *cláusula*.

– Definiciones

- Un **átomo** es una cadena de caracteres finita que comienza con una letra minúscula o con comillas simples. Por ejemplo: gato, ciudad, 'Sevilla', x12.
- Una **constante** es o bien un átomo o bien un número.
- Una **variable** es una cadena finita de caracteres alfanuméricos que comienza con una letra mayúscula. Por ejemplo: Nombre, Ciudad, X. Existe una variable especial llamada anónima (se nota por el carácter de subrayado "_") que no se instancia.
- Un **functor** se caracteriza por su nombre (que es un átomo) y por su número de argumentos. Un functor puede ser un operador (como el +) o un nombre de predicado.
- Cada predicado determina una función que va del dominio D (sobre el que está definido dicho predicado) en el conjunto de valores $\{verdadero, falso\}$.
- Un **término** es una variable o un átomo. Los términos compuestos o estructurados se construyen recursivamente aplicando functores a otros términos.
- Un **literal** es un predicado o su negación.
- Una **cláusula** es un conjunto de literales.
- Un **programa en Prolog** es una colección de cláusulas donde las variables son locales a cada cláusula, esto es, el ámbito de cada variable está limitado a la cláusula en la cual aparece.

– Tipos de Cláusulas

Existen tres clases de cláusulas:

1. *Reglas*: Una regla es una cláusula que consiste en una CABEZA, que es un literal positivo, y un CUERPO, formado por varios literales separados por comas (o por punto y coma). La cabeza y el cuerpo están separados por el símbolo ":-" (if). Una regla expresa una afirmación condicionada. Por ejemplo:

   abuelo(X,Y) :- padre(X,Z),padre(Z,Y)

   que indica que el abuelo de X es Y si el padre de X es Z y el padre de Z es Y.

   Nótese que las variables de la cabeza de la regla se consideran cuantificadas universalmente por defecto, mientras que las del cuerpo lo están existencialmente. La coma entre los literales representa la conjunción. Si se desea poner el cuerpo en forma disyuntiva, en lugar de la coma irá punto y coma (;).

2. *Hechos*: Un hecho es una cláusula formada por un único literal. Es una afirmación axiomática, esto es, que es verdad independientemente de cualquier condición. Por ejemplo:

   abuelo(juan,pedro).

3. *Objetivos*: Se llaman también cláusulas sin cabeza y representan las consultas. Por ejemplo:

   ?abuelo(X,pedro)

◇ ¿Cómo Calcula el Motor de Inferencia del Prolog?

Todo programa en Prolog consiste en una secuencia de hechos y reglas, que se denominan **cláusulas** y es, en definitiva, una declaración de las propiedades verificables por una serie de relaciones. Aunque desde el punto de vista lógico el orden en el que aparecen dichas cláusulas debería ser irrelevante, en el caso del Prolog, no lo es. De hecho el orden de las cláusulas es muy importante desde el punto de vista computacional.

El Prolog utiliza el Principio de Resolución de Robinson para resolver los objetivos que se le van presentando. Dado un objetivo a ser probado, el motor del Prolog busca

entre los hechos y reglas de su Base de Datos utilizando el método de **Búsqueda en Profundidad** para lograr unificar el actual objetivo con alguna o algunas cláusulas. La Base de Datos interna del Prolog se organiza como un **heap**, donde las cláusulas se alojan dinámicamente. Este almacenamiento funciona muy bien dada su extrema rapidez, pero limita mucho el número de cláusulas a considerar.

El proceso de computación del sistema de Prolog, puede resumirse en los siguientes pasos:

1. Dada una consulta u objetivo a ser probado, el sistema intenta resolver primero el sub-objetivo de más a la izquierda.
2. Para resolver un sub-objetivo, el sistema intenta emparejarlo con la cabeza de alguna de las cláusulas del programa.
3. Si el sub-objetivo en cuestión ha encajado satisfactoriamente con la cabeza de alguna cláusula, se genera una instancia de dicha cláusula dando valores a todas las variables que intervengan. Esta instancia es, entonces, sustituida por el sub-objetivo que la ha generado en el objetivo principal. A partir de ahí, el sistema selecciona el siguiente sub-objetivo a probar y vuelve a empezar.
4. Si el sistema no ha logrado emparejar el sub-objetivo actual con la cabeza de ninguna cláusula, éste intenta encontrar una solución volviendo atrás en el proceso; a esto se le conoce por *"backtraking"*.
5. Si todos los sub-objetivos del objetivo principal se resuelven, el sistema dará una respuesta *positiva*. Esta respuesta positiva tiene dos posibles formas:
   - Si el objetivo a verificar incluye variables, la respuesta dada por el sistema está constituída por los valores dados a dichas variables durante el proceso de prueba.
   - Si el objetivo no incluye variables, la respuesta dada es, simplemente, *"si"*.
6. Si no ha sido posible resolver todos los sub-objetivos del objetivo principal, el sistema responderá *"no"* (siempre y cuando la causa no sea un bucle infinito).
7. Si el sistema entra en un bucle infinito, la computación finalizará con un error (stack overflow).

◇ Conclusiones

A pesar de estar fundamentado en ella, el Prolog estándar difiere de la programación lógica pura en varios aspectos, entre los que cabe destacar:

* El uso de predicados construidos proceduralmente que permiten lectura y escritura de cláusulas, controlar los mecanismos de búsqueda, declarar estructuras de datos, variables y operadores aritméticos y utilidades para la depuración y traza de programas.
* El proceso de unificación de cláusulas para la demostración de teoremas difiere del que se sigue lógicamente en la resolución clásica, ya que se aplica de izquierda a derecha y de arriba a abajo.
* Prolog ofrece una evaluación desde el punto de vista de las demostraciones como efecto lateral de la evaluación en términos de la teoría de pruebas.
* La mayor parte de las implementaciones del Prolog no permite el tratamiento de la negación lógica de forma completa.

Como puede verse, el Prolog es una variante del cálculo de predicados de primer orden, lo que hace de él la herramienta ideal para la consulta y el manejo de información (que es uno de nuestros propósitos). Además de las características aquí mencionadas, el Prolog tiene características adicionales (sección 1.3) que aumentan su poder expresivo frente a los lenguajes de consulta clásicos (álgebra y cálculo básicamente). De hecho, es el lenguaje más utilizado en lo que a representación y programación lógicas se refiere y, en particular, en las implementaciones comerciales de Bases de Datos Lógicas e interfaces entre Bases de Datos Relacionales y entornos lógicos.

1.2 Las Bases de Datos Relacionales. Representación Lógica

Como es bien sabido, los estudios teóricos de Bases de Datos (BD) son la base fundamental para el desarrollo de buenos Sistemas de Gestión de BD (SGBD). La mayoría de estos estudios, están orientados al modelo relacional de BD, introducido por Codd en 1970 [23] cuya definición se apoya, fundamentalmente, en el formalismo de la lógica matemática. El propósito de este capítulo es ofrecer una visión de las BD relacionales (BDR) desde el punto de vista de la lógica (en particular, desde la lógica de primer orden), analizando qué ventajas se derivan de ello. Para ello, daremos, previamente, una breve revisión del modelo relacional de BD. Para finalizar, mostraremos cómo se han aprovechado las ventajas de ambas aproximaciones de las BD (la lógica y la relacional) para construir sistemas eficientes de gestión de datos.

1.2.1 Las Bases de Datos Relacionales

En esta sección vamos a tratar los conceptos más relevantes y elementos más representativos de los Sistemas de Bases de Datos Relacionales (SBDR). Estos conceptos tienen que ver con tres partes bien diferenciadas: la estructura de los datos, la integridad de los datos y la manipulación de los datos.

◇ Estructura e Integridad de los Datos

En relación con cómo están estructurados los datos, definiremos a continuación los siguientes conceptos:

Un **dominio** es un conjunto, normalmente finito, de valores. El producto cartesiano de una serie de dominios $D_1, D_2, \ldots, D_n$, que notaremos por $D_1 \times D_2 \times \ldots \times D_n$ es el conjunto de todas las tuplas $(x_1, x_2, \ldots, x_n)$ tales que para cualquier i, $i = 1, \ldots, n$ se verifica que $x_i \in D_i$. Se llama **relación** a cualquier subconjunto del producto cartesiano de dos o más dominios. Una **instancia** de BD es un conjunto finito de

relaciones finitas*. Se llama **cardinalidad** de una relación al número de tuplas que contiene. La **aridad** o **grado** de una relación $R \subset D_1 \times D_2 \times \ldots \times D_n$ es n, esto es, el número de dominios que intervienen en su definición.

Normalmente, una relación puede verse como una tabla de valores, en la que las columnas llevan asociado un nombre, que llamaremos **atributo**. Los valores de un atributo asociado a la columna i, pertenecen al dominio D_i. Una relación R de atributos $A_1, A_2, \ldots, A_n$ define lo que se llama un **esquema de relación**, y lo notaremos por $R(A_1, A_2, \ldots, A_n)$, de manera que una relación específica R_1, con un conjunto concreto de tuplas, se dice que es una *instancia* o *extensión* de dicho esquema.

No todas las instancias acordes con un determinado esquema son semánticamente válidas, esto es, tienen una interpretación coherente con la semántica que lleva asociada la BD. Por ello, se introduce un conjunto de restricciones, llamadas **restricciones de integridad**, que se asocian a cada esquema de relación, para asegurar que se preserva el significado coherente de los datos.

Por último, diremos que se llama **esquema de BD** a un conjunto de esquemas de relación junto con un conjunto de restricciones de integridad (RI). De este modo, una *instancia* o *estado* de una BD será, pues, un conjunto de instancias de relación (una por cada esquema de relación que intervenga en el esquema de la BD). Un *estado* de una BD será *válido*, si todas las instancias de relación que contiene verifican las RI impuestas.

Un estudio más profundo del modelo relacional de BD puede verse en [23] [26] [36] [69] [116].

◇ Manipulación de los Datos

En este apartado abordaremos el problema de la manipulación de las estructuras que aparecen en un SBDR. Esta manipulación se hace por medio de algún lenguaje formal de manejo de datos. En este sentido distinguiremos entre:

*Entendemos por relación finita aquella cuya extensión es finita, es decir, aquella que contiene un número finito de tuplas

– *Algebra Relacional*

– *Cálculo Relacional*

La diferencia esencial entre ambos es que mientras que el álgebra proporciona una colección de operadores y operaciones explícitas, el cálculo proporciona una notación para definir la relación resultante de una petición dada.

– Algebra Relacional

El Algebra Relacional (AR) es un lenguaje de consulta procedural, ya que el usuario da instrucciones al sistema para ejecutar una serie de operaciones en la BD, que calcularán los resultados deseados.

El AR consta de un conjunto de operadores que toman como entrada una o dos relaciones para dar como salida una nueva relación.

Los operadores básicos del álgebra relacional pueden agruparse de la siguiente manera:

* *Operación de asignación*: Es una operación especial que asigna el resultado de otras operaciones sobre relaciones a una nueva relación. Se incluye como operador para poder conservar los resultados.

* *Operaciones tradicionales sobre conjuntos*: Son Unión, Intersección, Diferencia y Producto Cartesiano. Para todas ellas, excepto el Producto Cartesiano, es necesario que las dos relaciones operando sean compatibles, esto es, que sean del mismo grado y que los i- ésimos atributos de las dos relaciones tengan el mismo dominio.

  – UNION: La unión de dos relaciones A y B da como resultado otra relación cuyas tuplas pertenecen a A, a B ó a ambas.

  – INTERSECCION: La intersección de dos relaciones A y B es el conjunto de tuplas que pertenecen a A y a B.

  – DIFERENCIA: La diferencia entre dos relaciones A y B es el conjunto de tuplas que pertenecen a A y no a B.

- PRODUCTO CARTESIANO EXTENDIDO: El producto cartesiano extendido de dos relaciones A y B es el conjunto de todas las tuplas tales que cada una de ellas es la concatenación de una tupla de A con una tupla de B. La concatenación de una tupla $a = (a_1, a_2, \ldots, a_m)$ con otra $b = (b_1, b_2, \ldots, b_m)$ es la tupla $t = (a_1, a_2, \ldots, a_m, b_{m+1}, b_{m+2}, \ldots, b_{m+n})$.

* *Operaciones especiales*: Son Selección, Proyección, Reunión y División.

- SELECCION: Este operador algebraico produce un subconjunto "horizontal" de una relación específica, es decir, el subconjunto de las tuplas de una relación para el cual se cumple un predicado dado, expresado como combinación booleana de términos (comparaciones).
- PROYECCION: Produce un subconjunto "vertical" de una relación dada, es decir, el subconjunto obtenido al seleccionar los atributos especificados en un orden también especificado y eliminar las tuplas duplicadas.
- REUNION: La reunión de la relación A sobre el atributo X con la relación B sobre el atributo Y da como resultado todas las tuplas t tales que t es la concatenación de una tupla de A con una de B tales que se verifica una condición determinada sobre X e Y. Una caso particular es la EQUI-REUNION, donde la condición exigida es X=Y.
- DIVISION: Divide una relación R de grado m+n entre una relación R' de grado n, produciendo una de grado m. El $(m+i)$-ésimo atributo de R y el i-ésimo de R' deben estar definidos sobre el mismo dominio.

– Cálculo Relacional

El Cálculo Relacional es, al contrario que el Algebra, un lenguaje no procedural de consulta, ya que el usuario sólo expresa aquella información que desea obtener de la BD, pero sin especificar qué operaciones han de realizarse sobre la BD para obtenerla.

Existen dos versiones del Cálculo Relacional: El Cálculo Relacional de Tuplas y el Cálculo Relacional de Dominios, que pasamos a describir a continuación.

Cálculo Relacional de Tuplas

Una expresión en el Cálculo Relacional de Tuplas (CRT), es una expresión de la forma:

$$\{P/P(t)\}$$

que representa el conjunto de todas las tuplas t que hacen verdadera la fórmula o predicado P. Usaremos $t[A]$ para denotar el valor que tiene la tupla t para el atributo A y $t \in R$ para denotar que la tupla t está en la relación R. En una misma fórmula pueden aparecer varias variables tupla. Se dice que una variable tupla es *libre* si no va cuantificada ni universal ni existencialmente.

Una fórmula del CRT se compone de *átomos*. Un átomo tiene una de las siguientes formas:

- $s \in R$, donde s es una variable tupla y R es una relación.
- $s[x]\Theta v[y]$, donde s y v son variables tupla, x es un atributo sobre el que s está definida, y es un atributo sobre el que v está definida, y Θ es un operador de comparación ($<$, $>$, $\leq$, $\geq$, $=$). En particular, se requiere que los atributos x e y tengan dominios cuyos valores puedan ser comparados por medio de Θ.
- $s[x]\Theta c$, donde s, x y Θ tienen el mismo significado que en el apartado anterior, y c es una constante del dominio del atributo x.

Las fórmulas se construyen a partir de los átomos usando las siguientes reglas:

- Un átomo es una fórmula.
- Si P_1 es una fórmula, entonces también los son (P_1) y $\neg(P_1)$.
- Si P_1 y P_2 son fórmulas, también los son $P_1 \wedge P_2$, $P_1 \vee P_2$ y $P_1 \Longrightarrow P_2$.
- Si $P_1(s)$ es una fórmula que contiene una variable de tupla libre s, entonces $\exists s \in R\ (P_1(s))$ y $\forall s \in R\ (P_1(s))$ también son fórmulas.

Existe un aspecto muy importante en el CRT que debe mencionarse. Tal y como se han definido las expresiones, podría generarse una relación infinita, como por ejemplo:

$$\{t/\neg(t \in R)\}$$

ya que existirá un número infinito de tuplas que no estén en la relación R. Dado que este tipo de expresiones no son deseables en absoluto, habrá que definir una restricción del CRT, para lo cual introducimos el concepto de *dominio de una fórmula* P. Intuitivamente, el dominio de P, que notaremos $dom(P)$, es el conjunto de todos los valores representados por P, esto es, aquellos valores mencionados en P así como los valores que aparecen en una tupla mencionada en P. Decimos que una *expresión* $\{t/P(t)\}$ *es segura* si todos los valores que aparecen en el resultado son valores de $dom(P)$.

El CRT restringido a expresiones seguras, es equivalente en poder expresivo al álgebra relacional. En [116] puede encontrarse la demostración formal de este hecho.

Cálculo Relacional de Dominios

Esta segunda forma del cálculo relacional usa variables de dominio, que toman valores del dominio de un atributo, en vez de valores de una tupla completa.

Una expresión del Cálculo Relacional de Dominios (CRD) es de la forma:

$$\{< x_1, x_2, \ldots, x_n > | P(x_1, x_2, \ldots, x_n)\}$$

donde $x_1, \ldots, x_n$ representan variables de dominio y P representa una fórmula compuesta por átomos. Un átomo en el CRD tiene una de las siguientes formas:

- $< x_1, x_2, \ldots, x_n > \in R$, donde R es una relación de n atributos y las x_i son variables o constantes de dominio.
- $x \Theta y$, donde x e y son variables de dominio y Θ es un operador de comparación. Es requisito indispensable que los atributos x e y tengan dominios que puedan compararse por medio de Θ.
- $x \Theta c$, donde x y Θ se definen como en el caso anterior, y c es una constante del dominio del atributo sobre el que se mueve x.

Las fórmulas se construyen de igual forma que en el CRT.

Respecto al problema de la seguridad en las expresiones para el CRD, diremos que una expresión es segura, si se cumplen las siguientes condiciones:

- Todos los valores que aparecen en tuplas de la expresión son valores de $dom(P)$.
- Cada subfórmula de la forma $\exists x\ (P_1(x))$ es verdadera si, y sólo si, existe un valor x en $dom(P_1)$ tal que $P_1(x))$ es verdadero.
- Cada subfórmula de la forma $\forall x\ (P_1(x))$ es verdadera si, y sólo si, $P_1(x))$ es verdadero para todos los valores x de $dom(P_1))$.

El propósito de estas reglas es asegurar que podemos probar las subfórmulas *"para todo"* y *"existe"* sin tener que probar infinitas posibilidades.

Por último, decir que el CRD restringido a expresiones seguras es equivalente al CRT (también restringido) y, por tanto, al AR.

Un sub-lenguaje del Cálculo Relacional para el manejo de datos, puede verse en [24].

Tanto el Algebra como el Cálculo (en sus dos versiones) Relacionales pueden verse ampliamente en [36] [69].

1.2.2 Representación Lógica de una BD Relacional

La lógica ofrece una base sólida en lo que respecta a la teoría de BD, especialmente para expresar consultas y para definir vistas y restricciones de integridad. Sin embargo, a la hora de caracterizar una instancia de una BD relacional por medio de lógica de primer orden, puede optarse por hacerlo desde dos puntos de vista distintos [58]: el de la teoría de modelos, esto es, la BD es una *interpretación* de una teoría especial de primer orden, y el de la teoría de pruebas, en el que la BD es considerada una *teoría relacional.*

El enfoque que se ha dado tradicionalmente a las BD ha sido el de la teoría de modelos, donde se considera que la definición del esquema de la BD es una teoría de

primer orden, invariante en el tiempo y que está constituida por las estructuras de datos y las restricciones de integridad. En este enfoque, un estado concreto de la BD es una interpretación que debe ser un modelo para la teoría. En este esquema la evaluación de una consulta consiste en calcular el valor de verdad del predicado que la constituye basándose en el estado actual de la BD.

Por el contrario, los amantes de la programación lógica prefieren el enfoque de la teoría de pruebas para BD. Los hechos (tuplas) y las reglas de deducción constituyen por sí mismos la teoría y las consultas son teoremas a ser probados a partir de la teoría, utilizando técnicas de demostración.

◇ Conceptos Previos

Definición 1.20 . *Un* **lenguaje** *de primer orden $L = (A, W)$ se dice* **relacional** *si y sólo si:*

1. *Hay un número finito de constantes en A pero $A \neq \emptyset$.*
2. *Hay un número finito de predicados en A.*
3. *A no contiene símbolos de función.*
4. *Entre los predicados de A hay que distinguir un predicado binario, el $=$.*
5. *Entre los predicados unarios se distingue un conjunto de ellos que llamamos* **tipos simples**. *Tales tipos simples, junto con su combinación booleana, nos ayudarán a modelizar el concepto de dominio de una relación.*

De entre todas las posibles interpretaciones dadas a un lenguaje relacional L, distinguimos un grupo de ellas, denominadas *interpretaciones relacionales*, que se caracterizan como sigue:

Definición 1.21 . *Sea $L = (A, W)$ un lenguaje relacional. Una* **interpretación** *$I = (U, K, P)$ se dice* **relacional** *si, y sólo si:*

- *K es el conjunto de las constantes de A sobre el dominio U, donde U es finito. Equivale al axioma de* mundo cerrado.
- *La extensión del predicado de igualdad es $P(=) = \{(d,d)/d \in U\}$. Equivale al axioma de* nombre único, *ya que $= (d,d')$ es falso para todos los elementos d y d' del dominio que sean distintos entre sí.*

Definición 1.22 *Teniendo en cuenta las dos definiciones anteriores, una* **BD Relacional** *es un triple (L, I, RI) donde:*

- *L es un lenguaje relacional.*
- *I es una interpretación relacional.*
- *RI es un conjunto de fbf de L que llamaremos restricciones de integridad.*

En particular, se requiere que, para cada predicado n-ario P, que no sea el de igualdad ni sea un tipo simple, L contenga, al menos, una fbf de la forma:

$$\forall\ X_1, X_2, \ldots, X_n,\ P(X_1, X_2, \ldots, X_n) \longrightarrow \tau_1(X_1) \wedge \tau_2(X_2) \wedge \ldots \wedge \tau_n(X_n)$$

donde los τ_i son tipos simples y se les llama dominios de P. Para cada predicado P que no sea un tipo simple, la extensión se llama relación.

◇ La BD como Interpretación Relacional

Sea BD_1 una instancia de una BD relacional. BD_1 estará formada por un conjunto de relaciones junto con las correspondientes restricciones de integridad. Sea U la unión de los dominios de todos los atributos que aparecen en los esquemas de las relaciones que conforman la BD $(R_i(A_{i_1}, A_{i_2}, \ldots, A_{i_n}))$. Con U así construido, la interpretación es un modelo para fórmulas de la forma:

$$\forall\ (X_1, X_2, \ldots, X_n),\ P(X_1, X_2, \ldots, X_n) \longrightarrow \tau_1(X_1) \wedge \tau_2(X_2) \wedge \ldots \wedge \tau_n(X_n)$$

que formalizan el concepto de *dominio* o esquema de una relación.

Si R es una relación de una instancia de BD llamada DB_1, la fórmula $R(a_1, a_2, \ldots, a_n)$ es verdad, si y sólo si, la tupla $< a_1, a_2, \ldots, a_n >$ pertenece a la relación R.

Una fórmula de la forma $\forall x,\ W(x)$ será verdadera en dicha interpretación, si y sólo si, para cada valor $d \in U$, $W(d)$ es verdadera.

Una fórmula de la forma $\exists x,\ W(x)$ será verdadera en dicha interpretación, si y sólo si, existe algún $d \in U$, tal que $W(d)$ es verdadera.

Si las restricciones de integridad se expresan como fórmulas bien formuladas del lenguaje, la instancia BD_1 de la BD será un estado válido, si y sólo si, cada restricción es verdadera en BD_1, esto es, si BD_1 es un modelo para el conjunto de restricciones de integridad.

◇ La BD como Teoría Relacional

Pasamos ahora a reformular el concepto de BD como una teoría. Para ello, es necesario introducir previamente algunos axiomas adicionales, lo que nos lleva a dar la siguiente definición:

Definición 1.23 . *Sea $R = (A, W)$ un lenguaje relacional. Una teoría de primer orden $T \subseteq W$ es una* **teoría relacional**, *si y sólo si se verifican las condiciones siguientes:*

1. *Si $c_1, c_2, \ldots, c_n$ son constantes de A, T contendrá la siguientes fbf:*

$$\forall x,\ = (x, c_1) \vee = (x, c_2) \vee \ldots \vee = (x, c_n)$$

   que se corresponde con el axioma de dominio cerrado.

2. *Si $c_1, c_2, \ldots, c_n$ son constantes de A, T contendrá la siguientes fbf:*

$$\forall i,\ \forall j,\ i \neq j,\ \neg = (c_i, c_j)$$

   que se corresponde con el axioma de nombre único.

3. T contiene los axiomas de igualdad*:*

- Reflexividad*:* $\forall x, =(x,x)$
- Conmutatividad*:* $\forall\ (x,y),\ =(x,y) \longrightarrow =(y,x)$
- Transitividad*:* $\forall\ (x,y,z),\ =(x,y) \wedge =(y,z) \longrightarrow =(x,z)$
- Principio de sustitución de términos iguales*: Para cada símbolo de predicado P de A, se incluirá la siguiente fbf:*

  $\forall\ (x_1,\ldots,x_n,y_1,\ldots,y_n),\ P(x_1,\ldots,x_n) \wedge =(x_1,y_1),\ldots,=(x_n,y_n)$
  $\longrightarrow P(y_1,\ldots,y_n)$

4. Sea $\Delta \subseteq T$ *el conjunto de fórmulas atómicas robustas de la teoría T (sin incluir las que involucran el predicado de igualdad), y sea* $C_p = \{(c_1,c_2,\ldots,c_n) \mid P(c_1,c_2,\ldots,c_n) \in \Delta\}$*, donde* C_p *recibe el nombre de* extensión de P en T*. En esta situación, para cada predicado P de A, que no sea el de igualdad, T contendrá la fbf:*

$$\forall\ (x_1,\ldots,x_n),\ \neg P(x_1,\ldots,x_n)$$

si $C_p = \{\ \}$ *(P no tiene estensión en T); sin embargo, si*

$$C_p = \{(c_1^1,\ldots,c_n^1),\ldots,(c_1^r,\ldots,c_n^r)\},$$

entonces habrá que incluir las siguientes fbf:

$$\forall\ (x_1,\ldots,x_n),\ P(x_1,\ldots,x_n) \longrightarrow$$

$$(=(x_1,c_1^1) \wedge,\ldots,\wedge =(x_n,c_n^1)) \vee \ldots \vee (=(x1,c_1^r) \wedge,\ldots,\wedge =(x_n,c_n^r))$$

llamado axioma de completitud para el predicado P *y*

$$\forall i,\ i \in \{1,2,\ldots,r\}\ P(c_1^i,\ldots,c_n^i)$$

que se llama extensión del predicado P*.*

5. Las únicas fbf de T son las que se construyen según los puntos anteriores.

Estas dos aproximaciones (la de teoría de pruebas y la de las interpretaciones) confluyen en el siguiente teorema.

Teorema 1.7 . *Sea $L = (A, W)$ un lenguaje relacional. Entonces:*

(*i*) *Si T es una teoría relacional de R, se verifica que T tiene un único modelo I, que es una interpretación relacional de R.*

(*ii*) *Si I es una interpretación relacional de R, entonces hay una teoría relacional T tal que I es el único modelo de T.*

La demostración correspondiente puede encontrarse en [102].

Notas:

1. La teoría de las BDR incorpora un conjunto de operadores relacionales como $<, >, <>, >=, \ldots$. Hemos considerado tan sólo el de igualdad, porque juega un papel muy importante en la teoría que vamos a desarrollar, pero resulta muy sencillo modificar la definición de BD para incorporar estos operadores y cualquier otro operador binario.

2. El concepto de restricción de integridad corresponde a lo que se llaman *restricciones o leyes estáticas*, que deben verificarse para cualquier estado de la BD.

– Especificación de Consultas

Las **consultas** se definen en función de un lenguaje relacional L =(A,W). Una consulta [102] [58] para L es cualquier expresión de la forma:

$$< X/\Phi,\ W(X) >$$

donde:

– X/Φ es $x_1/\phi_1, x_2/\phi_2, \ldots, x_n/\phi_n$, las x_i son variables de A y los ϕ_i son tipos simples o compuestos de A.

– $W(X)$ es una fbf cuyas únicas variables libres son las x_i.

Si DB=(L,I,RI) es una BDR entonces una consulta de L se dice que es aplicable a la BD.

Intuitivamente, la consulta $< X/\Phi,\ W(X) >$ denota el conjunto de tuplas constantes $C =< c_1, c_2 \ldots, c_n >$ tales que c_i es del tipo τ_i y tales que la BD satisface $W(C)$.

Las consultas *no aplicables* a DB pueden contener, o bien predicados que no pertenecen a A (y por tanto no tienen extensión en la teoría), o bien constantes que no son de A.

Algunos textos de interés en los que se analiza el poder expresivo de la lógica como lenguaje de consulta de BD son [72] [106] [59].

◇ Conclusiones

Como puede verse, la representación lógica de una BD además de ser sencilla, resulta fácilmente automatizable. Para llegar a implementar los pasos requeridos debemos elegir herramientas que nos permitan hacerlo de la forma más eficiente y directa posible. En este sentido, nos parece muy apropiada la elección de un lenguaje de programación lógica (en concreto Prolog) por su flexibilidad, posibilidades de ampliación del modelo (hacer deducción gracias a la definición de reglas) y porque permite una trascripción casi directa de la lógica de primer orden.

Hay que hacer notar, que el modelo relacional se identifica de forma muy natural en el contexto de las Interpretaciones, si bien algunos problemas clásicos (como el tratamiento de información disyuntiva e información nula) no tienen una buena solución en este enfoque.

Por otra parte, el enfoque de las Demostraciones nos conduce directamente al concepto de Base de Datos Lógica o Deductiva [57] [87], quedando una Base de Datos convencional como un caso particular de las mismas.

De los dos modelos (el de teoría de pruebas y el de las interpretaciones) no hay fundamentos sólidos para inclinarse por uno u otro. Más aún, dado un modelo de BD teórico *sin valores nulos*, éste puede transformarse en un conjunto de axiomas

de primer orden apropiados, tales que la teoría de primer orden resultante nos dé una caracterización en teoría de pruebas de la evaluación de una consulta y de las restricciones de integridad. Sin embargo, aunque ambas versiones son compatibles, la de teoría de pruebas es más rica y fructífera en el sentido de que permite cierto manejo de la información disyuntiva y valores nulos [102] (ambos aspectos se tratarán en el capítulo 3), evaluación de consultas sobre información incompleta, expresar (y obligar a que se verifiquen) restricciones de integridad, aplicar reglas de deducción y comparar distintos modelos de datos expresados en lógica. Todo ello, nos hace decantarnos por el enfoque de teoría de pruebas, ya que el objetivo que nos hemos propuesto es llegar a poder deducir conocimiento a partir de la información (tanto imprecisa como exacta) almacenada en una BD.

1.3 Acoplamiento entre Prolog y las BD Relacionales

El Prolog es el lenguaje lógico más popular y surgió como simplificación de técnicas generales de demostración de teoremas para ofrecer mayor eficiencia y programabilidad. De manera similar, el modelo relacional de BD nació como simplificación de complejos modelos jerárquicos o de red.

En la década de los 70 y comienzo de los 80 el uso tanto de Prolog como de las BD relacionales ha sido muy extendido, no sólo en el ámbito científico y académico, sino también en el ámbito comercial.

Además de las características mencionadas en el capítulo 1, el Prolog tiene características adicionales que aumentan su poder expresivo frente a los lenguajes de consulta clásicos (álgebra y cálculo básicamente).

Las características más destacables del Prolog como lenguaje de consulta son:

* Permite especificar una Base de Datos de forma no extensiva. Las reglas en Prolog ofrecen el medio de definir y evaluar relaciones derivadas o también llamadas intensivas (en contraposición a extensivas).
* La recursividad, que consiste en poder utilizar un predicado en su propia definición con lo que la cabeza de la regla creada para realizar tal definición estará contenida en el cuerpo de la misma. Gracias a la recursividad pueden resolverse problemas que no han podido ser resueltos con otros lenguajes estándar de consulta [19].
* Manejo de valores nulos en las reglas gracias a la variable anónima.
* El motor del Prolog es sensible al orden, esto es, es relevante el orden que tienen los hechos y las reglas en la Base de Datos de Prolog, lo cual permite al programador las posibilidad de ordenar los hechos y las reglas del modo más eficiente posible.
* Manejo de información negativa en la parte derecha de una regla.
* El backtraking o vuelta atrás, que es la capacidad del motor del Prolog de poder explorar el árbol creado durante la prueba de un objetivo realizando todas las

instanciaciones posibles de las variables. Si se desea evitar esto en algunos casos en que no conviene, se utiliza una primitiva, **cut**, para el control del backtraking.

Mientras que las dos primeras características son típicas de cualquier lenguaje de programación lógica, las restantes son propias del Prolog. En [89] puede verse que cualquier consulta expresada en SQL o en QBE puede ser fácilmente traducida a Prolog. Algunas posibilidades adicionales del Prolog en relación con las BD pueden verse en [79].

Analizando de cerca la programación lógica (Prolog) y los sistemas relacionales de BD, encontramos varios puntos en común, pero que difieren notablemente en su tratamiento:

* *Existencia de una BD*: Esta BD, en el caso de Prolog, es de tamaño relativamente pequeño, es mono-usuario y va almacenada en memoria principal. Su contenido es de reglas y hechos. En el caso de los sistemas de BD, se manejan grandes cantidades de datos almacenados en memoria masiva y que pueden ser compartidos (entorno multi-usuario).

* *La consulta de información*: Una consulta es el proceso por el que se extrae información relevante de la BD. En el entorno del Prolog, una consulta u objetivo se construye generando deducciones en cadena que combinan hechos y reglas con el fin de confirmar o refutar la afirmación inicial. En los sistemas de BD una consulta se determina encontrando el modo más eficiente de búsqueda en memoria masiva de la información requerida. (En [89] puede encontrarse una revisión sobre los lenguajes de consulta y su relación con el Prolog).

* *La especificación de restricciones*: Las restricciones definen las condiciones de correctitud para la BD. En los sistemas de BD la validación de restricciones es el proceso por el cual se preserva la correctitud de la BD evitando que en ella se almacenen datos incorrectos. En Prolog, las restricciones se expresan con reglas que se activan "automáticamente" cada vez que la BD es modificada.

En este sentido, la programación lógica ofrece un mayor poder (cuenta con la recursividad) para expresar consultas y restricciones comparado con el que ofrecen los

lenguajes de manipulación de datos de los sistemas de BD. Además, la programación lógica permite que la representación de consultas y restricciones sea homogénea y con los mismos mecanismos de inferencia. Por otro lado, esta última no ofrece la tecnología necesaria para manejar grandes cantidades de datos que pueden ser compartidos. Lo ideal, en este caso, es tratar de integrar ambos sistemas con el fin de explotar los beneficios que aporta cada uno de ellos.

El acoplamiento entre sistemas de BD clásicas y Prolog aparece en la literatura desde dos perspectivas muy distintas [19]:

1. Se traslada la BD a una representación lógica equivalente y los mecanismos de manipulación se traducen también a lógica de 1^{er} orden (Prolog). Se trabaja, pues, con herramientas y representación puramente lógicas.

2. Se aprovechan las ventajas de cada una de ellas construyendo sistemas de coordinación y manipulación de ambas herramientas conjuntamente. El proceso es más complicado pero más eficiente.

   Dentro de esta última perspectiva aparecen, a su vez, dos grandes modelos de arquitectura:

   - *Acoplamiento débil*: Sobre sistemas ya existentes se construye una nueva capa que los envuelve, pero manteniendo cada uno de ellos la estructura y comportamiento originales, es decir, los dos sistemas subyacentes se diferencian claramente. Este acoplamiento (ver fig. 1.1)se lleva a cabo en tiempo de carga del programa en Prolog y una vez que los hechos de la BD han sido cargados en memoria principal, la ejecución es independiente del sistema de BD.
   - *Acoplamiento fuerte*: Se construye un único sistema desde el principio haciendo uso, tanto de la tecnología de BD como de la programación lógica, interactuando ambas entre sí. En estos sistemas (ver fig. 1.2), la interacción entre el Prolog y el sistema de BD se lleva a cabo por el sistema de inferencia del Prolog. También se le llama acoplamiento dinámico porque las acciones se llevan a cabo cuando se está ejecutando el objetivo (goal) del Prolog, cuando se ejecuta una regla o cuando se va a realizar un emparejamiento (matching) con los predicados de la BD.

Con el método de acoplamiento débil, se construye un interface entre Prolog y un sistema de BD preservando cada uno su individualidad; el interface los constituyen procedimientos para trasladar los datos de la memoria masiva de la BD a la memoria principal en un formato adecuado al entorno de ejecución de los programas en Prolog.

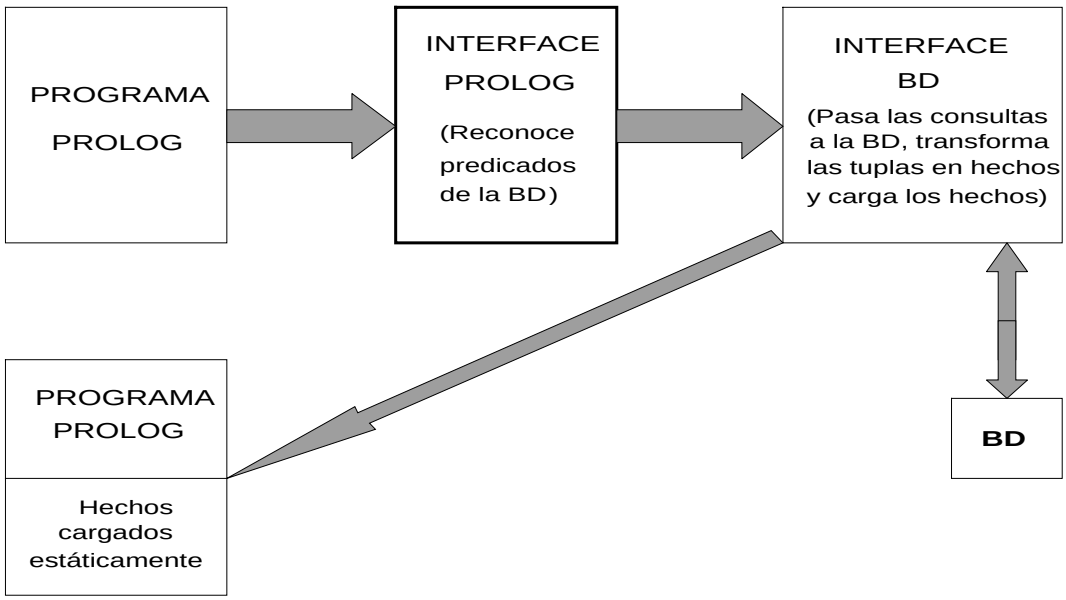


Figura 1.1. Arquitectura de un sistema de acoplamiento débil

En cualquiera de las dos aproximaciones, habrá que tener en cuenta que:

- Una gran parte de los predicados del programa en Prolog son predicados de la BD. Estos predicados se corresponden con las relaciones almacenadas en la BD y, por tanto, no deben aparecer nunca en la parte izquierda de una regla, esto es, no pueden redefinirse.
- Aunque para los SGBD el orden de los hechos en la BD no tiene relevancia, sí la tiene cuando se pasan a la BD del Prolog en memoria principal.
- Cuando dos o mas ocurrencias del mismo predicado de BD tienen las mismas variables acotadas y libres y en las mismas posiciones, hacen referencia a la misma fórmula (principio de sustitución de términos iguales).

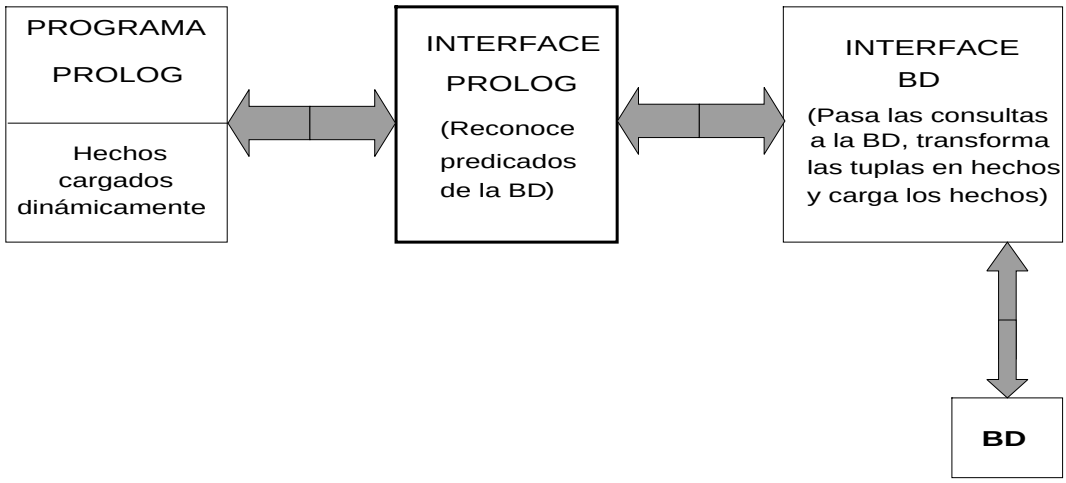


Figura 1.2. Arquitectura de un sistema de acoplamiento fuerte

1.3.1 Componentes de un Sistema de Acoplamiento Prolog-BD

Independientemente de como se lleve a cabo el acoplamiento entre el SGBD y el sistema Prolog, se distinguen cuatro subsistemas en su arquitectura:

1. **Motor de inferencia del Prolog**: que ejecuta programas en Prolog utilizando el mecanismo de inferencia estándar del Prolog.

2. **El interface del Prolog**: que es capaz de reconocer los predicados de la BD y tratarlos de forma especial. Puede ser totalmente transparente, sin intervención alguna del usuario. Un programa deberá acceder a la BD para reconocer los predicados. Este programa puede ser ejecutado cuando se carga el programa Prolog o cuando se desea probar un objetivo. También puede hacerse de forma no transparente al usuario de manera que el programador debe escribir explícitamente las consultas en el lenguaje del sistema de BD elegido. Existen sistemas con trasparencia intermedia de manera que el usuario debe dar la declaración de los

predicados de la BD; una vez declarados, éstos se manejan como los demás predicados salvo que, cuando son utilizados, el acceso a la BD es oculto. Interfaces de este tipo pueden verse en [81] y [79].

3. **Uno o mas interfaces de BD**: que facilitan y permiten la comunicación entre la BD y el motor de inferencia del Prolog. Está especialmente diseñado para acceder a la BD y extraer tuplas de ella. Existen varias posibilidades de hacerlo según el tipo de consulta realizada y según la forma de obtener los resultados de las mismas.

4. **El núcleo del sistema de BD**: es el software que lleva a cabo las modificaciones y consultas de datos en la BD de la memoria masiva. Este software puede darse a dos niveles:

   - *A bajo nivel*: Utilizando lenguajes procedurales que deben especificar las estrategias de acceso a los datos.
   - *A alto nivel*: Con lenguajes no procedurales que especifican las características a verificar por los datos a recuperar, mientras que el sistema se encarga de averiguar la mejor estrategia para llevarlo a cabo. El SQL es el más extendido de estos lenguajes.

1.3.2 Algunos Sistemas de Acoplamiento entre Prolog y BD

- **PRO-SQL**

  Este sistema fue desarrollado por el Centro de Investigaciones de IBM, para acoplar Prolog al sistema SQL/DS. Su descripción completa puede encontrarse en [21]. La principal característica es su total ausencia de transparencia, lo que obliga a que los programas sean escritos por personas conocedoras tanto del Prolog como del SQL. Aún así, se utiliza un predicado especial llamado "SQL" que se usa para incluir sentencias que se van a ejecutar en el entorno SQL de la BD. El predicado tienen la forma:

  $$\text{SQL}(<\textit{Sentencia SQL}>)$$

  Esta sentencia puede ser de cualquier tipo. Por ejemplo:

SQL('Select NOMBRE,SALARIO from EMPLEADOS where SALARIO>50').

y el efecto que tiene es añadir, en forma de hechos de Prolog, todas las tuplas que satisfacen esta petición a la BD de memoria principal del entorno Prolog.

La ejecución de predicados recursivos se hace por medio de llamadas repetidas al predicado SQL. PROSQL es ejecutable bajo el sistema operativo VM/CMS. EL motor del Prolog y el SQL/DS son ejecutados como dos sistemas independientes (máquinas virtuales) bajo la misma máquina física.

– **EDUCE**

EDUCE es uno de los grandes proyectos de integración de Programación Lógica y BD llevados a cabo por el European Computer-Industry Research Center (ECRC). Este sistema soporta tanto acoplamiento débil como fuerte. EDUCE se usa como núcleo del Prolog-KB, un sistema basado en el conocimiento soportado por un modelo semántico de datos. Todos estos sistemas utilizan el sistema INGRES de BD y el QUEL como lenguaje de consulta.

EDUCE soporta varios lenguajes de usuario y estrategias de implementación. Desde el punto de vista lingüístico, ofrece dos tipos de lenguaje: lenguaje débil no procedural y un lenguaje fuerte, parecido al Prolog en estilo.

Respecto a las estrategias de implementación, puede utilizar la orientada a conjuntos de tuplas y la orientada a una tupla cada vez. Pueden entremezclarse dando lugar a estrategias híbridas.

El *lenguaje débil* resulta algo complicado e incluye algunos predicados construidos a propósito y con una sintáxis y semántica especiales; tal es el caso de los predicados : *"query"*, que lleva como argumento una consulta en lenguaje QUEL y *"retrieve"*, que permite predicados con variables compartidas, con los que se pierde la transparencia.

El *lenguaje fuerte* es totalmente transparente y los predicados de la BD pueden entremezclarse con otros predicados, permitiéndose la recursividad.

El optimizador del EDUCE es capaz de seleccionar la estrategia más adecuada en cada caso.

Para más detalles sobre el sistema, ver [8] [9].

– **STEAM**

El interface STEAM, ha diseñado como resultado de un proyecto Esprit por el Laboratorio de Investigación de Philips (Bruselas) y por Onera-Cert (Toulouse).

Este interface trata de reconocer grandes *"conjunciones básicas"* [20] o unidades de interacción con la BD. Existen algoritmos para detectar las conjunciones básicas. El sistema de BD usado es el INGRES y el lenguaje de consulta el QUEL. Este interface incluye algoritmos para construir conjunciones básicas a partir de objetivos de Prolog.

Dado que establece un acoplamiento débil entre ambos sistemas, todas las interacciones con la BD se realizan antes de que se active el motor del Prolog. Este interface soporta disyunción, negación, recursividad y el predicado especial "cut".

Una visión más amplia puede encontrarse en [34].

– **BERMUDA**

Este prototipo (ver [66]) ha sido desarrollado en la Universidad de Wisconsin. El modelo se centra en los aspectos referentes a la arquitectura del sistema, ya que es capaz de emplear múltiples interfaces de BD a la misma vez.

Este paralelismo se lleva a cabo gracias a dos mecanismos básicamente:

a) El motor de inferencia del Prolog se activa tan pronto como se obtiene la primera tupla resultado de la BD; a partir de ahí, los interfaces de la BD y del Prolog continuan trabajando de forma asíncrona. Esto permite al motor del Prolog ir resolviendo otras consultas antes de que haya concluido la actual.

b) Cada uno de los interfaces de BD que interviene no obtiene el resultado completo de una consulta de la propia BD, sino que coge unos cuantos bloques (páginas) y los almacena en la memoria principal; antes de que el motor del Prolog llegue al final de la página actual, el interface de la BD se activa de nuevo para sacar la siguiente o siguientes páginas. Esta política de lectura adelantada reduce la cantidad de datos a almacenar en memoria principal así como el tiempo de espera del motor del Prolog.

Tras una consulta, las tuplas de la relación resultado se almacenan en un fichero externo y recuerda qué consultas han sido ya resueltas para no volver a calcu-

larlas en el caso en que se soliciten de nuevo (ya que bastará con leer el fichero correspondiente).

En particular, tiene un interface con Prolog, llamado BERMUDA AGENT, que recibe peticiones de consulta del motor del Prolog y las pasa al correspondiente interface de BD. Existe un número máximo de peticiones a atender; si se excede dicho número las peticiones son puestas en cola.

– **CGW y PRIMO**

CGW [16] [18] es un sistema que fue desarrollado en la Universidad de Stanford dentro del proyecto KBMS. Una de las principales características de CGW es que reduce los accesos a la BD pasando, tanto las consultas como los datos a la memoria principal. Utiliza un interface de bajo nivel para interactuar con la BD y controlar el acceso de las páginas de datos. Un meta- intérprete juega el papel de interface de Prolog, controlando el emparejamiento de predicados de la BD.

El interface de la BD recupera una página de datos de la BD y la añade a la BD residente de Prolog. Si una consulta ha sido completada, añade, además, el predicado *"consultado(dbp)"* (queried). La principal limitación de esta aproximación es el emparejamiento individual de cada predicado de la BD en lugar de buscar conjunciones básicas.

El modelo CGW ha sido el que se ha seguido en la realización del prototipo PRIMO [17], siendo principales características de su diseño la portabilidad, la modularidad y la transparencia.

La *portabilidad* permite que el interface de PRIMO actúe entre cualquier Prolog y cualquier sistema de BD siempre que el primero admita llamadas al sistema operativo y el segundo soporte el SQL. Para conseguirlo posee un analizador de programas capaz de transformar el programa en Prolog inicial en un programa en una versión en un Prolog específico para PRIMO, todo ello sin intervención del usuario.

El sistema PRIMO usa acoplamiento tanto fuerte como débil. El interface de la BD opera a alto nivel usando el lenguaje SQL, perdiéndose el control directo sobre los mecanismos de acceso a las páginas.

– **QUINTUS-PROLOG**

Este interface, desarrollado por *Quintus Computer Systems Inc.* [101], es ejecutable en estaciones de trabajo SUN y en máquinas VAX/UNIX. Este interface opera a dos niveles:

- *A nivel de relación*: devolviendo una tupla de las relaciones implicadas cada vez.
- *A nivel de vistas*: en el que una regla completa de Prolog se traduce a una consulta a la BD, incluyendo el "join" como operador de agregación. A este nivel sólo se puede trabajar cuando el cuerpo de las reglas incluye sólo predicados de la BD.

Este interface no almacena en memoria principal ni los datos ni las consultas; la comunicación tiene lugar a través de variables compartidas que toman valores determinados. Soporta, además, inserción y borrado de hechos de la BD y la creación de *vistas*.

◇ Conclusiones

A la vista de las grandes ventajas que aporta la programación lógica como herramienta de representación y manipulación de información, muchos autores se han planteado la creación de sistemas mixtos en los que se aprovechen las posibilidades tanto del modelo relacional como de los entornos lógicos de programación. En este sentido, han aparecido muchos modelos, tanto teóricos como implementados, que tienen en común una arquitectura básica a partir de la cual se ha construido cada sistema con sus características particulares.

Una vez sentadas estas bases, nos proponemos ampliar el modelo lógico subyacente de manera que pueda dar cabida a diferentes tipos de información, esto es, información disyuntiva, valores desconocidos, valores no aplicables, valores imprecisos,... así como derivar información adicional no almacenada explícitamente.

Capítulo 2

Preliminares II: Conceptos Basados en la Teoría de Conjuntos Difusos

En este capítulo, presentamos los conceptos más relevantes en cuanto a los fundamentos de las bases de datos lógicas difusas, esto es, la **lógica difusa** como extensión de la lógica clásica (secc 1.1.), por un lado, y las **bases de datos difusas** como extensión de las clásicas (secc 1.2), por otro.

Así mismo, estableceremos una nomenclatura homogénea que será empleada en los sucesivos capítulos.

2.1 Introducción a la Lógica Difusa

2.1.1 Introducción

La Lógica, es la ciencia que estudia los principios formales del razonamiento. Según esta definición, la Lógica Difusa trata de los principios formales del razonamiento aproximado, en el que, el razonamiento preciso, es un caso particular. Esto es, la característica esencial de la Lógica Difusa es que, al contrario de lo que sucede en la clásica, permite

modelizar de alguna manera el razonamiento impreciso que, por otro lado, es la base del pensamiento humano. La idea básica es poder dar una respuesta aproximada a una pregunta, en función de unos hechos previamente almacenados y que pueden ser inexactos, incompletos o poco fiables. Los grandes inconvenientes que plantea la Lógica clásica para llevar a cabo este tipo de *razonamiento* son:

1. En Lógica Bivaluada, una proposición p es verdadera o falsa. En Lógica Multivaluada, una proposición puede ser verdadera, falsa, o tener un valor de verdad intermedio de entre los de un conjunto finito de posibles valores de verdad. En Lógica Difusa, se permite que los valores de verdad sean subconjuntos difusos definidos, generalmente, en el intervalo $[0, 1]$.

2. En Lógica Bivaluada, los predicados han de ser, necesariamente, precisos, en el sentido de que no pueden describir subconjuntos difusos sobre el universo del discurso. En Lógica Difusa, los predicados pueden ser precisos (*madre,mortal,hombre,...*) o bien difusos (*alto, cansado, guapo,...*).

3. En las Lógicas Bivaluada y Multivaluada, sólo se permiten dos cuantificadores: *alguno* y *todos*. En Lógica Difusa, pueden utilizarse numerosos cuantificadores (*la mayoría, pocos, muchos, normalmente, alrededor de 4,...*), que son vistos como números difusos que expresan la cardinalidad de algún conjunto difuso.

4. La Lógica Difusa permite representar y manipular modificadores (difusos o no) de predicados, tales como: *ligeramente, mucho, un poco, bastante,....*

5. En Lógica Bivaluada, una proposición puede ser cualificada asociándole un valor de verdad verdadero o falso, mediante un operador modal como *posible* o *necesario* o bien mediante un operador intensional como *creo*, *sé*, *pienso*,... En Lógica Difusa, hay tres tipos de cualificación de predicados: *Cualificación de la verdad*, por ejemplo *"No es muy cierto que Pedro sea alto"*, *Cualificación de la probabilidad*, por ejemplo *"Es poco probable que Pedro sea alto"* y *Cualificación de la posibilidad*, por ejemplo *"Es prácticamente imposible que Pedro sea alto"*.

En resumen, la Lógica clásica resulta restrictiva en cuanto a que:

– No ofrece los mecanismos adecuados para representar simbólicamente sentencias cuyo significado es impreciso.

– No ofrece un mecanismo de inferencia propicio para llevar a cabo el razonamiento aproximado.

El objetivo de esta sección es exponer de manera clara los fundamentos de las lógicas multivaluada y difusa y de la teoría de la posibilidad, ya que son las herramientas fundamentales de nuestro trabajo.

2.1.2 Lógica Multivaluada

La lógica multivaluada, permite ampliar la lógica clásica en el sentido de que los valores de verdad asociados a las fórmulas no se limitan a 0-falso o 1-verdadero. En particular, la lógica multivaluada de Lukasiewicz [3] permite como valor de verdad de una fórmula, cualquier valor real del intervalo $[0, 1]$. En esta lógica, si $v(P)$ es el valor de verdad asociado a P y P y Q son proposiciones, entonces se definen los operadores clásicos de la siguiente forma:

$v(\neg P) = 1 - v(P)$

$v(P \ and \ Q) = v(P) \wedge V(Q)$

$v(P \ or \ Q) = v(P) \vee V(Q)$

$v(P \longrightarrow Q) = 1 \wedge (1 - v(P) + v(Q))$

$v(P \equiv Q) = v((P \longrightarrow Q) \ and \ (Q \longrightarrow P))$

Este sistema es completamente funcional en cuanto a los valores de verdad ser refiere y preserva la propiedad de que el valor de verdad de $P \longrightarrow Q$ es siempre 1 cuando $v(Q) \geq v(P)$. Esta lógica es el fundamento o la base lógica que soporta la lógica difusa definida por Zadeh y que presentamos a continuación.

2.1.3 Lógica Difusa y Teoría de la Posibilidad

La Lógica Difusa* está fundamentada en la teoría de subconjuntos difusos, que apareció sobre 1965 y cuyo autor es L.A. Zadeh [126], [128], [130], [132], y está basada en el hecho de que cualquier propiedad P determina un subconjunto S_P de los objetos que la verifican.

La lógica difusa puede considerarse una extensión de la lógica multivaluada, aunque los objetivos de una y otra son, en principio, bastante distintos. Dado que la lógica difusa está orientada a modelos de razonamiento aproximado, los razonamientos que se realizan no prestan excesiva atención al rigor o a la exactitud en las respuestas, ya que el concepto de *verdad* pasa a ser gradual. Esta posee, adicionalmente, un gran poder expresivo ya que contiene como casos particulares, a las lógicas bivaluada, multivaluada, posibilística y probabilística.

En este marco, la teoría de la probabilidad tiene una gran limitación, derivada del hecho de estar basada en lógica binaria, lo que obliga a que los predicados sean completamente precisos. De esta manera, los sucesos que vengan definidos por medio de predicados vagos (joven, alto, inteligente,...) y de cuantificadores lingüísticos (la mayoría, pocos, algunos,...) no se podrán manejar por medio de modelos basados en la teoría de la probabilidad. Además, dicha teoría no ofrece facilidades para llevar a cabo inferencia a partir de premisas vagas.

El objetivo de la lógica difusa es, pues, dar un marco más general, que permita el tratamiento tanto de la vaguedad como de la incertidumbre. Para ello, introduce la imprecisión a dos niveles:

- La introducción de predicados vagos en el mismo lenguaje.

- El tratamiento de los predicados metalingüísticos *verdadero* y *falso* como difusos, en particular que los valores de verdad asignados a los predicados son subconjuntos difusos del intervalo $[0, 1]$.

*Entre los autores españoles sobre la materia está acuñado el término mixto *Lógica Fuzzy*.

◇ Introducción a la Teoría de Conjuntos Difusos

En esta sección daremos un breve repaso del concepto de conjunto difuso. El lector interesado puede consultar [114], [46], [125].

De acuerdo con la nueva concepción de la noción de conjunto, toda propiedad determina un conjunto: el conjunto de todas las cosas que satisfacen dicha propiedad. Si se identifican tales propiedades con funciones de dominio U (donde U es algún universo de objetos) en $\{0,1\}$, entonces tales propiedades y los subconjuntos que determinan son indistinguibles, esto es, cualquier propiedad P determina un conjunto $S_P = \{u \in U \ : \ P(u) = 1\}$, y al contrario, cualquier subconjunto $S \subset U$ genera una propiedad (su función de pertenencia) P_S que viene dada por $P_S(u) = 1$ *si y solo si* $u \in S$. Estas mismas consideraciones pueden también hacerse cuando los conjuntos considerados son *conjuntos difusos*.

Veamos primero de forma intuitiva qué se entiende por **Conjunto Difuso**. Supongamos que la propiedad que estamos considerando es la *altura* de las personas. Respecto a esta propiedad sabemos que no es siempre posible calificar a una persona de alta o no alta (de hecho, en muchos casos tal afirmación es subjetiva). Este hecho nos conduce de forma muy natural hacia la *Lógica Tri-valuada*. Sin embargo, la Lógica Difusa va más allá, considerando que los predicados son funciones de U en $[0,1]$. Obsérvese, que si P es ahora una propiedad difusa (por ejemplo, *altura*), el conjunto que dicha propiedad determina tendrá la siguiente forma:

$$\{< u, i > : \ P(u) = i, \ u \in U, \ i \in [0,1]\}$$

de manera que los conjuntos difusos están determinados por tales predicados vagos que son los que constituyen su función de pertenencia. Dado un conjunto difuso S, notaremos su función de pertenencia por μ_S. De esta forma, un conjunto difuso puede verse como un conjunto de objetos cada uno de los cuales lleva asociado un grado de pertenencia.

A continuación, pasamos a definir algunos conceptos básicos en relación con la teoría de conjuntos difusos, con la idea de tener una nomenclatura común en lo sucesivo.

– *Universo de discurso*: Es un conjunto clásico U sobre el que está definida la función

de pertenencia.

- *Valor crisp*: Es cualquier elemento del universo del discurso.
- *Relación difusa*: Es un conjunto difuso definido sobre un producto cartesiano de universos de discurso. A continuación vamos a definir dos relaciones de este tipo, por su importancia para nosotros.

Definición 2.1 .

Diremos que una relación difusa R definida sobre $U \times U$ es una **relación de semejanza** *si, y sólo si, verifica las siguientes propiedades [127]:*

(i) Reflexiva*:* $R(x,x) = 1, \ \forall \ x \in U$

(ii) Simétrica*:* $R(x,y) = R(y,x), \ \forall \ x,y \in U$

Definición 2.2 .

Diremos que una relación difusa R definida sobre $U \times U$ es una **relación de similitud**, *si, y sólo si, verifica [127]:*

(i) R *es una* relación de semejanza

(ii) R *verifica la propiedad* Max–Min Transiva, *esto es:*

$$R(x,z) \geq max_{y \in U}\{min(R(x,y), R(y,z))\}$$

- El *soporte* de un conjunto difuso es el conjunto de elementos del universo que tienen un grado de pertencia al conjunto mayor que cero, esto es, el conjunto S_F definido como:

$$S_F = \{x \in U \ \mid \ \mu_F(x) > 0\}$$

- El *núcleo* de un conjunto difuso, K_F, es el conjunto de valores del universo que pertenecen al conjunto con grado 1, esto es:

$$K_F = \{x \in U \ \mid \ \mu_F(x) = 1\}$$

– Un conjunto difuso se dice *normalizado* si, y sólo si

$$\exists x \in U \mid \mu_F(x) = 1$$

esto es, si el núcleo no es el conjunto vacío.

– El α-*Corte* de un conjunto difuso se define como

$$F_\alpha = \{x \in U \mid \mu_F(x) \geq \alpha\}$$

– *Etiqueta lingüística*: es una distribución de posibilidad a la que se asigna un identificador. Este concepto fue desarrollado por Zadeh y puede verse en [128], [129], [130]. Cuando el universo del discurso es contínuo, suele utilizarse una representación trapezoidal para este tipo de datos (ver [40]).

– Operadores sobre Conjuntos Difusos

Una vez definido el concepto de conjunto difuso, hay que ver, en primer lugar, cómo se redefinen los operadores clásicos sobre conjuntos (unión, intersección, inclusión, complemento,...) para el caso de los conjuntos difusos. Los operadores usuales de la teoría de conjuntos clásica pueden extenderse a la de conjuntos difusos de varias formas, haciendo uso de T−normas, T−conormas y funciones de negación. Los operadores más usuales son los siguientes:

– **Intersección**: Sean A y B dos subconjuntos difusos de U. Entonces la función de pertenencia de la intersección de A y B se define como sigue:

$$\mu_{A \cap B}(x) = Min\{\mu_A(x), \mu_B(x)\}$$

– **Unión**: Sean A y B dos subconjuntos difusos de U. Entonces, la función de pertenencia de la unión de A y B se definie como sigue:

$$\mu_{A \cup B}(x) = Max\{\mu_A(x), \mu_B(x)\}$$

– **Complemento**: Sean A y B dos subconjuntos difusos de U. Decimos que A y B son complementarios si, y solo si, $\forall x,\ x \in U,\ \mu_B(x) = 1 - \mu_A(x)$

Este conjunto de operadores constituyen el modelo **Max–Min**, que preserva todas las propiedades de un álgebra booleana, *excepto* las leyes de *no contradicción* y del *tercero excluído*, esto es, no se verifica que:

$$A \cap \neg A = \emptyset$$

$$A \cup \neg A = U$$

donde $\emptyset$ es el conjunto dado por $\mu_\emptyset(x) = 0,\ \forall x \in U$.

◇ Introducción a la Teoría de la Posibilidad

Por su estrecha relación con la lógica difusa, introducimos, a continuación, algunos conceptos preliminares de gran importancia.

– Concepto de Medida Difusa.

Consideremos un conjunto de sucesos asociados a un cuerpo de conocimiento impreciso e incierto y consideremos que estos sucesos son subconjuntos de un conjunto de referencia Ω llamado *suceso seguro*. Identificaremos al conjunto vacío, $\emptyset$, con el llamado *suceso imposible*. Supondremos que existe un número real asociado a cada suceso $A \subseteq \Omega$, lo notamos $g(A)$, que mide la confianza que se puede tener en la ocurrencia del suceso A, teniendo en cuenta el estado actual de conocimiento. Por convenio, $g(A)$ crece conforme lo hace la confianza en dicho suceso, de manera que si A es un suceso seguro, $g(A) = 1$, mientras que si es un suceso imposible, $g(A) = 0$. En particular, $g(\Omega) = 1$ y $g(\emptyset) = 0$.

El axioma más débil que puede concebirse para asegurar una mínima coherencia de g es que sea monótona con respecto a la inclusión, esto es:

$$A \subseteq B \Longrightarrow g(A) \geq g(B)$$

que significa que si A implica B, uno puede tener en en B al menos tanta confianza como en A.

Cuando Ω es un conjunto de referencia infinito, pueden introducirse axiomas de continuidad como el siguiente:

Sean $\{A_1, A_2, \ldots, A_n\}$ una secuencia anidada de conjuntos verificando $A_0 \subseteq A_1 \subseteq \ldots \subseteq A_n$ o bien $A_n \subseteq A_{n-1} \subseteq \ldots \subseteq A_0$, entonces:

$$\lim_{n\to\infty} g(A_n) = g(\lim_{n\to\infty} A_n)$$

Toda medida difusa debe verificar este axioma para alguna de las dos secuencias especificadas (creciente o decreciente).

Estas medidas fueron propuestas por Sugeno [112] para evaluar la incertidumbre.

– Medidas de Posibilidad y Necesidad

Como consecuencia del axioma de monotonía, se verfican, de forma inmediata, las siguientes expresiones:

$$\forall A, B \subseteq \Omega, \quad g(A \cup B) \geq max\{g(A), g(B)\}$$

$$g(A \cap B) \geq min\{g(A), g(B)\}$$

Como caso particular, encontramos medidas, que notaremos por Π, que verifican que:

$$\forall A, B, \quad \Pi(A \cup B) = max\{\Pi(A), \Pi(B)\}$$

llamadas **medidas de posibilidad**.

Suponiendo que $E \subseteq \Omega$ es un suceso seguro, puede definirse fácilmente una función en $\{0, 1\}$ tal que:

$$\Pi(A) = 1 \; si \; A \cap E \neq \emptyset$$

$$\Pi(A) = 0 \; en \; otro \; caso$$

Es inmediato ver que, en este contexto, $\Pi(A) = 1$ significa que A es *posible*. En particular, si A y $\overline{A}$ son dos sucesos contradictorios, entonces se cumple:

$$max\{\Pi(A), \Pi(\overline{A})\} = 1$$

que significa que, de dos sucesos contradictorios, uno de ellos, al menos, es completamente posible.

Cuando el conjunto Ω es finito, toda medida de posibilidad Π puede definirse por medio de los valores de los singletons de Ω, de manera que:

$$\forall A, \quad \Pi(A) = sup\{\pi(\omega) \mid \omega \in A\}$$

donde $\pi(\omega) = \Pi(\{\omega\})$ y π es una función de Ω en $[0, 1]$ llamada **distribución de posibilidad**. Esta función está normalizada, en el sentido de que $\exists\omega, \; \pi(\omega) = 1$, ya que $\Pi(\Omega) = 1$.

Otro caso particular de medida, que notaremos N, es el que se obtiene cuando se cumple:

$$\forall A, B \;\; N(A \cap B) = min\{N(A), N(B)\})$$

A esta clase de medidas se les llama **medidas de necesidad**. De manera análoga al caso anterior, puede construirse una función en $\{0, 1\}$ en base a un suceso seguro, como sigue:

$$N(A) = 1 \; si \; E \subseteq A$$

$$N(A) = 0 \; en \; otro \; caso$$

Es inmediato ver que $N(A) = 1$ significa que A es seguro.

Así pues, la teoría de la posibilidad utiliza dos medidas para representar la incertidumbre: la *posibilidad* y la *necesidad*. Ambas medidas deben verificar que:

$$N(A) = 1 - \Pi(\neg A) \tag{2.1}$$

donde $N(A)$ expresa hasta qué punto puede considerarse que la proposición o suceso A es necesariamente cierta, y significa que una proposición es más cierta cuánto menos posibilidad haya de que se dé lo contrario.

A continuación, presentamos algunos de los *axiomas y relaciones de la teoría de la posibilidad*. Un estudio más amplio de la teoría de la posibilidad puede verse en [46], [125], [77].

$$\Pi(\Omega) = 1 \tag{2.2}$$

$$N(\emptyset) = 0 \tag{2.3}$$

y

$$\forall(A, B), \quad \Pi(A \cup B) = max\{\Pi(A), \Pi(B)\} \tag{2.4}$$

De dichas expresiones puede deducirse que:

$$N(\Omega) = 1 \tag{2.5}$$

$$\Pi(\emptyset) = 0 \tag{2.6}$$

y que

$$\forall(A, B), \quad N(A \cap B) = min\{N(A), N(B)\} \tag{2.7}$$

Si, además, las proposiciones A y B no son vagas, y dado que se verifican las expresiones:

$$max\{\Pi(A), \Pi(\neg A)\} = 1 \tag{2.8}$$

$$min\{N(A), N(\neg A)\} = 0 \tag{2.9}$$

se pueden deducir las siguientes:

$$\forall A,\ \Pi(A) \geq N(A) \tag{2.10}$$

$$N(A) > 0 \Longrightarrow \Pi(A) = 1 \tag{2.11}$$

$$\Pi(A) < 1 \Longrightarrow N(A) = 0 \tag{2.12}$$

$$N(A) + N(\neg A) \leq 1 \tag{2.13}$$

$$\Pi(A) + \Pi(\neg A) \geq 1 \tag{2.14}$$

– Proposiciones Vagas

Sea X *es alto* una proposición vaga. Un aspecto fundamental de las proposiciones vagas, es que no se da un valor concreto a la variable X, sino que definen una distribución de posibilidad de los valores de X que asocia un número del intervalo $[0, 1]$ a cada uno de los posibles valores de X, conocido que *"X es alto"*. Esta equivalencia puede expresarse de manera simbólica como:

$$"X\ es\ alto" \ \equiv\ \Pi_X = ALTO$$

o, lo que es igual,

$$\Pi_X(x) = \mu_{ALTO}(x)$$

En particular, un hecho preciso corresponde al caso en el que la distribución de posibilidad toma valor cero en todo el dominio, excepto en un punto en el que toma valor uno.

Una distribución de posibilidad sobre un conjunto discreto de valores del universo la notaremos de la forma:

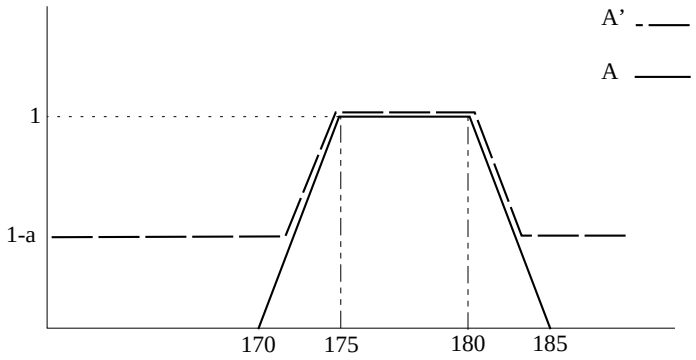


Figura 2.1. Representación del conjunto difuso **alto** con certeza a.

$$u_1/\pi_1 + u_2/\pi_2 + \ldots + u_n/\pi_n$$

donde los u_i son los elementos del dominio y los π_i los correspondientes valores de posibilidad.

En muchas ocasiones, la vaguedad y la incertidumbre se dan simultáneamente, porque cualquier hecho, ya sea preciso o difuso, puede ser incierto. Un hecho es incierto si no hay seguridad de que el valor real de la variable X esté en A. Para expresar esta incertidumbre, es necesario un valor α del intervalo $[0,1]$ que llamaremos *grado de certeza o de necesidad*, y significa que existe una posibilidad $1-\alpha$ de que X tome su valor fuera de A. Si tenemos la proposición $p =$"X *es* A" tal que $N(p) = \alpha < 1$, donde α es el grado de necesidad de p, esta proposición puede reescribirse como :

$$p' = X \ es \ A'$$

donde p' sigue siendo vaga pero no incierta. En esta situación, el conjunto A' tendría la forma que aparece en la figura 2.1.

Posibilidad y necesidad también pueden utilizarse sobre **proposiciones vagas**. Supongamos la proposición

$$p = X \ es \ A$$

donde A es un conjunto difuso con universo de referencia U caracterizado por su función de pertenencia μ_A. Sea B otro conjunto difuso de universo de referencia U y caracterizado también por su función de pertenencia. En esta situación, la posibilidad de $X \ es \ B$ cuando se sabe que $X \ es \ A$, viene dada por la expresión:

$$\Pi(B) = sup(A \cap B)$$

que, utilizando el mínimo como operador para la intersección [132], nos lleva a la expresión:

$$\Pi(B) = sup_x\{min(\mu_A(x), \mu_B(x))\} \tag{2.15}$$

de la que se obtiene fácilmente que

$$N(B) = 1 - \Pi(\neg A) = inf_x\{max(\mu_B(x), 1 - \mu_A(x))\}$$

En particular, si B es *crisp* entonces se cumple que:

$$\Pi(B) = sup_{x \in B}(\mu_A(x))$$

Es importante destacar que las expresiones 2.1, 2.4 y 2.7, usando los operadores del modelo Max−Min, se siguen verificando aunque los predicados involucrados sean vagos. Sin embargo, deja de cumplirse la expresión

$$max\{\Pi(A), \Pi(\neg A)\} = 1 \tag{2.16}$$

ya que deja de tener sentido la ley del tercero excluído.

◇ Representación del Conocimiento en Lógica Difusa

Como ya comentábamos en la introducción, una de las principales, si no la más importante, característica de la lógica difusa es el hecho de que los valores de verdad asignados a los predicados son subconjuntos difusos del intervalo $[0,1]$. Sin embargo, Zadeh no admite cualquier subconjunto de estas características, sino que se restringe a un conjunto de ellos a los que denomina *valores de verdad lingüísticos*. Esto es, el conjunto de posibles valores de verdad (V_V) es de la forma: $V_V = \{$verdadero, falso, no verdadero, muy verdadero, mas o menos verdadero, casi verdadero, no muy falso,...$\}$, de manera que cada elemento del conjunto es generado a partir del subconjunto difuso denotado como *verdadero*, esto es, si μ_V es la función de pertenencia para el subconjunto *verdadero*, algunas de las funciones de pertenencia para otros elementos de V_V podrían ser las siguientes:

$$\mu_{falso}(u) = \mu_V(1-u)$$
$$\mu_{no\ verdad}(u) = 1 - \mu_V(u)$$
$$\mu_{muy\ verdad}(u) = (\mu_V(u))^2$$
$$\mu_{casi\ verdad}(u) = (\mu_V(u))^{1/2}$$

de manera que, una vez fijados el significado de *verdadero* y las reglas de cálculo, ya se tiene el significado del resto de los valores de verdad.

La otra idea básica que subyace en todo el formalismo de la lógica difusa, es considerar una proposición p como una colección de restricciones elásticas $\{C_1, C_2, \ldots, C_k\}$ que restringen los valores de un conjunto de variables $\{X_1, X_2, \ldots, X_n\}$. Por lo general, tanto las restricciones como las variables restringidas que aparecen en una proposición suelen venir dadas de manera implícita y habrá que hacerlas explícitas para poder representar su significado. Esto se hace representando la proposición p en lo que se llama la *forma canónica*, como sigue:

$$p \text{ pasa a la forma } X \ es \ A$$

donde A es una predicado difuso. Dicho de otra forma, la forma canónica de p implica que la distribución de posibilidad de la variable X es A ($\Pi_X = A$), o lo que es lo mismo,

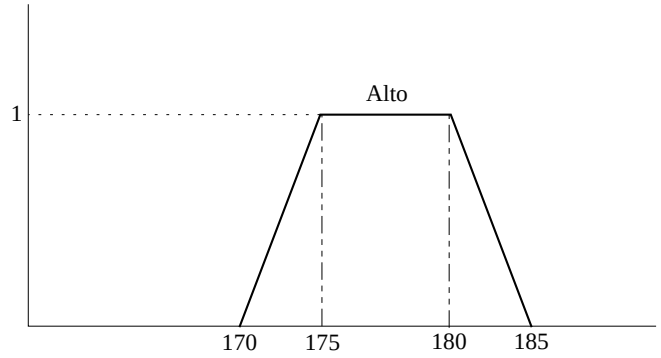


Figura 2.2. Posible representación del conjunto difuso **alto**.

que $Poss\{X = u\} = \mu_A(u),\ u \in U$, donde μ_A es la función de pertenencia de A y $Poss\{X = u\}$ es la posibilidad de que la variable X tome el valor u.

Veamos un ejemplo. Supongamos que nuestra proposición p es la siguiente: *"Pedro es alto"*. En este caso, la variable X es $Altura(Pedro)$ y A es la relación difusa $ALTO$, cuya representación podría ser la de la figura 2.2, con lo que su forma canónica quedaría expresada como:

$$Altura(Pedro)\ es\ ALTO$$

De esta expresión se deduce que $Poss\{Altura(Pedro) = u\} = \mu_{ALTO}(u)$, donde μ_{ALTO} es la función de pertenencia de la relación difusa $ALTO$ y $\mu_{ALTO}(u)$ es el grado en el que el valor numérico u verifica la restricción inducida por $ALTO$.

Cuando p es una proposición condicional, su forma canónica se expresaría:

$$Y\ es\ B\ si\ X\ es\ A$$

esto es, la proposición p induce una distribución de posibilidad condicionada de Y dado

X, que notaremos por $\Pi_{(Y/X)}$. En el marco de la lógica difusa, esta distribución de posibilidad condicionada puede venir definida de numerosas formas [46],[77], [49].

Otra posibilidad, es que la proposición p sea de la forma

$$Q\ A\ son\ B$$

donde Q es un cuantificador difuso y A y B son predicados difusos. Una proposición de este tipo sería *"La mayoría de las personas altas no están muy gruesas"*. Obsérvese que también en este caso tendremos una variable X, que será la proporción de B que son también A, y Q es la correspondiente restricción elástica. Más formalmente, si U es un conjunto finito $\{u_1, u_2, \ldots, u_m\}$, la proporción de B en A se define como la $\sum -suma\ relativa$, esto es:

$$\sum Suma(B/A) = \frac{\sum_j \mu_A(u_j) \wedge \mu_B(u_j)}{\sum_j \mu_A(u_j)} \quad , j = 1, \ldots, m. \tag{2.17}$$

donde $\mu_A(u_j)$ y $\mu_B(u_j)$ son los grados de pertenencia de u_j a los conjuntos A y B respectivamente.

2.1.4 Mecanismos de Inferencia

En lógica difusa, las reglas de inferencia* (ver [134],[135],[136]) pueden clasificarse de muchas formas. Una de estas clasificaciones distingue entre:

- **Reglas Categóricas**: Son aquéllas que no contienen cuantificadores difusos.
- **Reglas Cuantificadas**: Contienen algún tipo de cuantificador difuso. De entre las reglas cuantificadas, se distinguen las llamadas **Reglas Disposicionales**, en las que alguna de las premisas contiene el cuantificador difuso *normalmente*, ya sea de manera implícita o explícita.

A continuación pasamos a ver los mecanismos de inferencia más comunes tanto de las reglas categóricas como de las cuantificadas.

*También llamadas *silogismos*.

◇ Mecanismos de Inferencia con Reglas Categóricas

– Principio de Generalización

Una regla básica de inferencia, que llamaremos *principio de generalización* (*entailment rule*), es la que se muestra a continuación:

$$\frac{\begin{array}{l} X \ es \ A \\ A \ \subset \ B \end{array}}{X \ es \ B} \tag{2.18}$$

esto es, si una variable X toma sus valores en un conjunto A y A está contenido en un conjunto B, puede generalizarse la primera afirmación diciendo que X toma sus valores del conjunto B.

El principio de generalización tal como lo hemos presentado, es una regla categórica. Si alguna de las premisas fuera cuantificada por *normalmente*, obtendríamos el *principio de generalización disposicional*, por ejemplo:

$$\frac{\begin{array}{l} normalmente \ (X \ es \ A) \\ A \ \subset \ B \end{array}}{normalmente \ (X \ es \ B} \tag{2.19}$$

Desde otro punto de vista distinto, el principio de generalización de la Lógica Difusa puede considerarse como un *principio de herencia* de propiedades, esto es, si la primera premisa se interpreta como que X *tiene la propiedad* A, entoces la conclusión debe interpretarse como X *tiene la propiedad* B donde B es cualquier conjunto que contiene a A.

– Regla de Conjunción

Esta regla tiene la siguiente forma:

$$\frac{\begin{array}{l} X \ es \ A \\ X \ es \ B \end{array}}{X \ es \ A \cap B} \tag{2.20}$$

donde $A \cap B$ viene dada por:

$$\mu_{A \cap B}(u) = \mu_A(u) \wedge \mu_B(u), \ u \subset U$$

– Regla del Producto Cartesiano

Esta regla tiene la siguiente forma:

$$\frac{\begin{array}{l} X \ es \ A \\ Y \ es \ B \end{array}}{(X, Y) \ es \ A \times B} \tag{2.21}$$

donde (X, Y) es una variable bidimensional y $A \times B$ viene dada por:

$$\mu_{A \times B}(u, v) = \mu_A(u) \wedge \mu_B(u), \ u \subset U, \ v \subset V$$

– Regla de Proyección

Esta regla tiene la siguiente forma:

$$\frac{(X, Y) \ es \ R}{X \ es \ R_X} \tag{2.22}$$

donde R_X es la proyección de la relación binaria R sobre el dominio de X, y viene dada por:

$$\mu_{R_X}(u) = sup_v \mu_R(u, v), \ u \subset U, \ v \subset V$$

– Regla Composicional

Esta regla tiene la siguiente forma:

$$\frac{\begin{array}{l} X \ es \ A \\ (X,Y) \ es \ R \end{array}}{Y \ es \ A \circ R} \tag{2.23}$$

donde $A \circ R$ es la composición de la relación A con la relación binaria R, y que viene dada por:

$$\mu_{A \circ R}(v) = sup_u(\mu_A(u) \wedge \mu_R(u,v))$$

– Modus Ponens Generalizado

Puede verse como un caso particular de la Regla Composicional de Inferencia, y tiene la siguiente forma:

$$\frac{\begin{array}{l} X \ es \ A \\ Y \ es \ C \ si \ X \ es \ B \end{array}}{Y \ es \ A \circ (\neg B \oplus C)} \tag{2.24}$$

donde $\neg B \oplus C$ viene dada por:

$$\mu_{\neg B \oplus C}(u,v) = 1 \vee (1 - \mu_B(u) + \mu_C(v))$$

Una característica muy importante que posee el modus ponens generalizado que no la posee el modus ponens de la lógica clásica, es que el antecedente de la segunda premisa y la primera premisa no tienen por qué ser idénticos.

Si se aplica el modus ponens generalizado a reglas disposicionales, este mecanismo de inferencia puede expresarse de la siguiente forma:

$$\frac{\begin{array}{l} normalmente\ (X\ es\ A) \\ normalmente\ (Si\ X\ es\ A,\ Y\ es\ B\ ) \end{array}}{normalmente^2\ (Y\ es\ B)} \tag{2.25}$$

donde se han hecho coincidir la premisa primera y el antecedente por simplicidad.

Un caso particular de este mecanismo de inferencia se da cuando se parte de una aproximación funcional de las reglas, en la que la regla se ve como una función de $\{p, \neg p\}$ en $\{q, \neg q\}$ que relaciona p y q por medio de una medida de posibilidad (necesidad) condicional, que notaremos $\Pi(q/p)$ ($N(q/p)$). Esta posibilidad (necesidad) condicional mide la posibilidad de deducir q a partir de p y se define como la mayor solución de la ecuación

$$\Pi(p \wedge x) = TN(\Pi(x/p), \Pi(p)), \quad x \in \{q, \neg q\} \tag{2.26}$$

que puede encontrarse en [47] y donde TN es una T-Norma.

Nótese que este mecanismo de inferencia contiene al modus ponens clásico como caso particular cuando p y $p \longrightarrow q$ son verdaderas. A partir de las distribuciones $\Pi(q/p)$ y $\Pi(p)$ o $\Pi(q)$ (o sus correspondientes expresiones en términos de necesidad) pueden obtenerse los siguientes patrones de inferencia [77]:

$$\frac{\begin{array}{ll} \Pi(q/p) & \geq \alpha \\ \Pi(p) & \geq \beta \end{array}}{\Pi(q) \quad \geq TN(\alpha, \beta)} \tag{2.27}$$

$$\frac{\begin{array}{ll} N(q/p) & \geq \alpha \\ N(p) & \geq \beta \end{array}}{N(q) \quad \geq min(\alpha, \beta)} \tag{2.28}$$

En el marco de inferencia bajo incertidumbre de naturaleza probabilística, Suppes obtuvo ya en el año 1966 [113] patrones de inferencia análogos, modelizando las reglas por medio de probabilidades condicionales, esto es:

$$\frac{\begin{array}{ll} P(q/p) & \geq \alpha \\ P(p) & \geq \beta \end{array}}{P(q) \quad \geq \alpha * \beta} \tag{2.29}$$

◇ Mecanismos de Inferencia con Reglas Cuantificadas

– Silogismo de Intersección-Producto

Este silogismo puede ser expresado mediante la siguiente regla de inferencia:

$$\frac{\begin{array}{l} Q_1 \ A \ son \ B \\ Q_2 \ (A \ y \ B) \ son \ C \end{array}}{(Q_1 \otimes \ Q_2) \ A \ son \ (B \ y \ C)} \tag{2.30}$$

donde Q_1 y Q_2 son cuantificadores difusos, A y B son predicados difusos y $Q_1 \otimes Q_2$ es el producto de los números difusos Q_1 y Q_2.

Dado que la intersección de B y C está contenida en C, de la regla anterior se sigue la siguiente:

$$\frac{\begin{array}{l} Q_1 \ A \ son \ B \\ Q_2 \ (A \ y \ B) \ son \ C \end{array}}{\geq (Q_1 \otimes \ Q_2) \ A \ son \ C)} \tag{2.31}$$

En particular, si los cuantificadores Q_1 y Q_2 son monótonos crecientes, entonces $\geq (Q_1 \otimes Q_2) = Q_1 \otimes Q_2$. Si, además, B es un subconjunto de A, entoces $A \ y \ B = B$ y la regla anterior pasa a ser una regla encadenada de la forma:

$$\frac{\begin{array}{l} Q_1 \ A \ son \ B \\ Q_2 \ B \ son \ C \end{array}}{(Q_1 \otimes \ Q_2) \ A \ son \ C} \tag{2.32}$$

Veamos un ejemplo. Si tenemos las proposiciones:

La mayoría de los estudiantes son solteros

y

La mayoría de los solteros son jóvenes

según el esquema de inferencia 2.32, podremos deducir que

La mayoria^2 *de los estudiantes son jóvenes*

donde *mayoria*^{2} puede expresarse en función de *mayoria* según lo indica la figura 2.3.

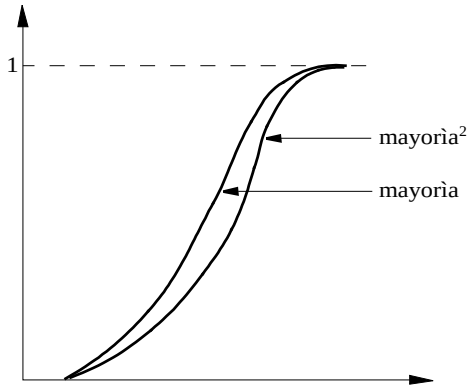


Figura 2.3. Representación gráfica de las funciones *mayoria* y *mayoria*^{2}

– Silogismo de Conjunción en el Consecuente

Este silogismo tiene la siguiente forma:

$$\begin{array}{l} Q_1\ A\ son\ B \\ Q_2\ A\ son\ C \\ \hline Q\ A\ son\ (B\ y\ C) \end{array} \qquad (2.33)$$

donde

$$0 \overset{\circ}{\vee} (Q_1 \oplus Q_2 \ominus 1) \leq Q \leq Q_1 \overset{\circ}{\wedge} Q_2$$

y $\leq$, $\overset{\circ}{\vee}$, $\overset{\circ}{\wedge}$, $\oplus$ y $\ominus$ son las correspondientes extensiones difusas de los operadores clásicos $\leq$, $\vee$, $\wedge$, $+$ y $-$ [128].

– Silogismo de Conjunción en el Antecedente

Un problema más complejo es el que se presenta cuando hay una combinación conjuntiva de antecedentes. Este hecho puede representarse de la forma siguiente:

$$\frac{\begin{array}{l} Q_1\ A\ son\ C \\ Q_2\ B\ son\ C \end{array}}{Q\ (A\ y\ B)\ son\ C} \tag{2.34}$$

Sin imponer ninguna restricción sobre los valores Q_1, Q_2, A, B y C, no se puede afirmar nada acerca de Q. La restricción que se suele imponer por los sistemas expertos (MYCIN [11]) es la independencia condicional de A y B.

2.2 Las Bases de Datos Difusas

2.2.1 Introducción

Los modelos tradicionales de BD que aparecen en la literatura, sólo son capaces de manejar y representar datos ideales, y suponen que la información en ellas almacenada es exacta, correcta y está bien definida. Sin embargo, en el mundo real existe una gran cantidad y variedad de datos cuya naturaleza no permite que sean formulados de forma precisa. Esto significa que la información que se posee sobre un atributo, existe o no existe, pero no se permite ningún grado de incertidumbre o de imprecisión respecto de la misma, es decir, no se pueden representar ni tratar informaciones del tipo *"Jorge es más o menos alto"*.

Tampoco se contempla la obtención de información en términos imprecisos a partir de la que se encuentra en la BD almacenada en forma precisa; así, por ejemplo, no se ofrecen mecanismos para obtener, de una atributo que almacene información sobre las alturas de una población, aquellos individuos que sean *"altos"*.

En este capítulo, nos vamos a centrar en el estudio de las nuevas técnicas aparecidas en el campo de las BD que permiten la representación y el tratamiento de este tipo de información, y haremos un análisis de los principales modelos existentes en la literatura.

2.2.2 Características Generales de un Modelo de Base de Datos Relacional Difusa

Los aspectos más importantes de la información vaga que manejamos habitualmente son: la **incertidumbre** y la **imprecisión**. La primera se deriva de apreciaciones realizadas sobre nuestra observación de la realidad, como por ejemplo: *"Me parece que Juan es profesor de Universidad"*. La imprecisión, sin embargo, se manifiesta a través del enunciado de conceptos que no están bien diferenciados o definidos, como por ejemplo: *"Luis es alto"*, donde alto es un concepto cuya semántica puede variar según quien lo emplee y que involucra un conjunto de valores posibles para la altura de Luis. Es habitual que en nuestras conversaciones se mezclen ambos aspectos y es obvio que

los enunciados mostrados en los dos ejemplos proporcionan un grado de información sobre el mundo real. En algunos casos, la información que proporcionan puede resultar suficiente, pero en otros puede no serlo.

Todas estas consideraciones nos conducen a plantear la conveniencia de que el sistema satisfaga las siguientes características:

1. Debe proporcionar los mecanismos adecuados para poder representar información difusa en todas sus vertientes. No es misión del sistema el dotar a esa información de ninguna semántica particular. Esta tarea ha de realizarla el usuario a la hora de diseñar la aplicación particular que resuelva su problema. Sin embargo sí debe proporcionar el entorno adecuado para representar dicha semántica, por tanto,

2. Deberá ofrecer el marco adecuado para representar y almacenar el significado de la información difusa que albergue. La información en sí misma no aporta ninguna utilidad si no se dispone de mecanismos adecuados para recuperarla y para operar con ella.

3. Tendrá que facilitar un conjunto mínimo de operadores para recuperar y tratar la información atendiendo a la naturaleza difusa de la misma.

4. Deberá satisfacer los requisitos del modelo relacional en la mayor medida posible. Como vimos, el modelo relacional presenta una serie de características que no se cumplen al cien por cien en ningún sistema de BD comercial. Por ello, es lógico pensar que los modelos difusos de BDR presenten, así mismo, distinta adecuación a dicho modelo. En este sentido, parece sensato exigir que haya algún atributo cuyos valores sean perfectamente conocidos con el fin de poder identificar unívocamente las tuplas y poder definir el concepto de llave primaria.

2.2.3 Principales Modelos de Bases de Datos Difusas

En general, cualquiera de los modelos que aparece en la literatura cae dentro de alguno de estos tres enfoques:

- El Modelo Relacional Difuso

- El Modelo de Relaciones de Similitud.
- El Modelo Relacional Posibilístico.

◇ Modelo Relacional Difuso

Este modelo extiende el modelo clásico mediante la adición de un grado a cada tupla. Este grado representa hasta qué punto la tupla pertenece a la relación en la que se encuentra y es un valor numérico del intervalo [0,1].

Más formalmente, una BD Difusa sería un conjunto de relaciones en el que cada una de ellas es una relación difusa caracterizada por una función de pertenencia como sigue:

$$BD = \{R_1, R_2, \ldots, R_m\}$$

$$\mu_{R_j} : U_1 \times U_2 \times \ldots \times U_n \longrightarrow [0,1] \quad j = 1 \ldots m$$

siendo U_i el dominio del i-ésimo atributo de la relación y $\times$ el producto cartesiano.

Aunque se han construido sistemas basados en este modelo [132], [6], éste presenta el inconveniente de que cada tupla asume su carácter difuso de forma global sin que se pueda determinar la aportación difusa particular de cada uno de los atributos que la constituyen.

◇ Modelo de Relaciones de Similitud

Este modelo, propuesto por Buckles y Petry ([12], [13], [2]), define una Relación Difusa como un subconjunto del producto cartesiano $2^{D_1} \times 2^{D_2} \times \ldots \times 2^{D_m}$, donde 2^{D_i} es cualquier elemento del conjunto de las partes del dominio ($P(D_i)$).

Los dominios permitidos para los atributos son los siguientes:

- Un conjunto finito de escalares. Ej. negro,rojo,verde.
- Un conjunto finito de números. Ej. 15,16,17.

	Rubio	**Cast.**	**Pelir.**	**Moreno**
Rubio	1	0.6	0.4	0
Cast.	0.6	1	0.5	0.6
Pelir.	0.4	0.5	1	0.4
Moreno	0	0.6	0.4	1

Tabla 2.1. Relaciones de similitud para los valores de *Color de Pelo*.

– Un conjunto de números difusos o etiquetas. Ej. bajo,medio,alto [13].

La forma de manejar y representar la incertidumbre es a través de las Relaciones de Similitud [127] sobre dominios escalares o numéricos y se hace manifiesta en el planteamiento de consultas, donde el usuario podrá imponer un umbral de similitud a partir del cual todas las tuplas que sean α-similares serán fundidas en una sola (Se construyen clases de equivalencia).

En esta línea están los modelos presentados en [108], [109] y [104]. En este último se proponen distintas relaciones entre los valores (no de similitud).

Veamos un ejemplo. Supongamos que tenemos un atributo definido sobre el dominio color de pelo compuesto por los valores: rubio, castaño, pelirrojo, moreno, y que sobre los valores tenemos definida la relación de similitud que aparece en la tabla 2.1.

Los valores tomados en la relación de similitud deberán ser suministrados por el usuario y ofrecen la medida en que los valores del dominio se parecen según la apreciación del observador.

En el proceso de consulta, el usuario pregunta por aquellas tuplas que satisfacen una determinada condición con unos determinados umbrales de similitud. El sistema tomaría dicha condición y, mediante las relaciones de similitud definidas, agruparía en clases de equivalencia las tuplas de la relación de partida. Los operadores clásicos del álgebra relacional operan sobre dichas clases de equivalencia.

Aclaramos estas ideas con un ejemplo sencillo. Supongamos la tabla de datos personales 2.2 y supongamos que se plantea la siguiente consulta:

Nombre	**Edad**	**Altura**	**ColorPelo**
Luis	16	{Alto,Muy-Alto}	Castaño
José	17	Bajo	Moreno
Laura	15	Muy-Bajo	Rubio
María	{15,16}	Media	Pelirrojo

Tabla 2.2. Tabla de Datos Personales

Nombre	**Edad**	**Altura**	**ColorPelo**
{Luis,José,Laura}	{15,16,17}	{Alto,Muy-Alto,Bajo, Muy-Bajo}	{Rubio,Castaño, Moreno}
María	{15,16}	Media	Pelirrojo

Tabla 2.3.

"Dame el nombre y la altura de aquellas personas cuyo color de pelo sea similar a castaño en grado mínimo 0.6"

En primer lugar, se crearían las clases de equivalencia a partir de las relaciones de similitud y de los umbrales de similaridad exigidos, lo cual daría como resultado la tabla 2.3

Obsérvese que los umbrales impuestos han generado dos clases de equivalencia o tuplas. De este conjunto de clases de equivalencia se seleccionarán aquéllas que superen el umbral exigido y se proyectarán sobre los campos Nombre y Altura, dando como resultado la tabla 2.4.

Los principales *inconvenientes* de este modelo son:

Nombre	**Altura**
{Luis,José,Laura}	{Muy-Alto,Alto,Bajo,Muy-Bajo}

Tabla 2.4.

* La definición de tupla no se corresponde con el concepto de item atómico de información, ya que una tupla en el sentido de Buckles-Petry equivale a varias unidades de conocimiento o interpretaciones (consecuencia de ello es que no se puede definir el concepto de llave primaria sobre las relaciones).

* En cuanto a la utilización de números difusos, ésta resulta poco intuitiva ya que las relaciones entre los mismos se establecería por medio de tablas de relación de similitud construidas expresamente para el modelo.

* La utilización del álgebra relacional con clases de equivalencia produce en algunos casos resultados confusos. Piénsese por ejemplo en actuar a través del operador REUNION sobre relaciones de equivalencia ...¿qué tuplas reunimos?.

Sin embargo, podemos señalar una serie de *ventajas*:

* El empleo de relaciones de similitud proporciona una herramienta adecuada e intuitiva para representar la imprecisión de conceptos de tipo cualitativo.

* El uso de umbrales para cada uno de los atributos implicados en una consulta. Con ello se proporciona al usuario un mecanismo con el que establecer qué atributos han de satisfacer una propiedad determinada y con qué precisión.

◇ Modelos Relacionales Posibilísticos

Estos modelos representan la imprecisión en los datos por medio del uso de distribuciones de posibilidad [132]. La estructura general de una base de datos en este modelo es como sigue:

La base de datos está constituida por un conjunto de relaciones

$$BD = \{R_1, R_2, \ldots, R_m\}$$

donde cada relación verifica:

$$R_j(U_1) \times (U_2) \times \ldots \times (U_n) \quad j = 1...m$$

(U_i) representa la familia de todas las distribuciones de posibilidad en el dominio U_i y $\times$ es el *Producto Cartesiano*.

Bajo esta definición hay muchos modelos (que difieren entre sí en la representación de las distribuciones de posibilidad, manejo de información desconocida o nula, el planteamiento de la consulta, operadores de selección, etc...) entre los que se encuentran los propuestos por Fukami-Umano [51] [118], Prade-Testemale [99], Zemankova-Kandel [138], ... cuyas características más destacadas pasamos a comentar.

– Modelo Fukami-Umano

Este modelo [51] admite como valores de atributo:

- Distribuciones de posibilidad
- Los valores unknown, undefined y NULL con la siguiente semántica:

$$unknown = \{1/u \ : \ u \in U\}$$

$$undefined = \{0/u \ : \ u \in U\}$$

$$Null = \{1/unknown, 1/undefined\}$$

los cuales serán procesados de forma específica por la implementación del sistema.

Veamos, mediante un ejemplo, como se representa la información. Para ello, supongamos la existencia de la tabla 2.5.

El nombre de los hijos de Ricardo, {Julia,Ana}, es una distribución de posibilidad indicando que el valor puede ser Julia o Ana (si fueran las dos aparecerían ambos nombres en tuplas separadas con el nombre del padre duplicado, como ocurre con Susana). La edad de Jaime es la distribución de posibilidad que se corresponde con la etiqueta joven, la cual podría venir definida por:

$$joven = \{0.3/15, 0.6/16, 0.8/17, 1/18, 1/19, 1/20, 1/21,$$

$$1/22, 1/23, 0.9/24, 0.8/25, 0.7/26, 0.5/27, 0.3/28, 0.1/29\}$$

Nombre	**Edad**	**Hijos**
Tomás	23	Alfredo
Susana	35	Juan
Susana	35	Miguel
Ricardo	40	{Julia,Ana}
Jaime	joven	Unknown
Victor	Unknown	Undefined
Elena	{50,51}	Null

Tabla 2.5.

El modelo resuelve la consulta dividiendo en tres subconjuntos las relaciones implicadas en la misma: el primero contiene las tuplas que satisfacen la consulta claramente, el segundo agrupa las tuplas que, posiblemente, la satisfacen y el tercero, contiene las tuplas que claramente no la satisfacen.

Veamos como se haría la consulta *"Dame las personas que tienen más de 25 años"*.

$$satisfacen\ claramente = \{1/Susana, 1/Ricardo, 1/Elena\}$$
$$satisfacen\ posiblemente = \{1/Jaime, 1/Victor\}$$
$$no\ satisfacen\ claramente = \{1/Tomas\}$$

Como la consulta en sí no es difusa, los grados de satisfacción son 0 ó 1.

El modelo asocia a cada item involucrado en la consulta, un par de verdad constituido por una pareja de elementos $< c, t >$. El factor c toma el valor T para aquellos items que reflejan certeza, y toma el valor P, para aquellos que reflejan posibilidad. El factor t asocia el valor de verdad en el sentido de la lógica multivaluada y toma su valor en el intervalo [0,1].

Para evaluar una proposición que contenga predicados difusos, se actuaría como sigue:

Sean $t_1, t_2, \ldots, t_n$ los valores de verdad evaluados para todas las combinaciones posibles de elementos u_x y u_y obtenidas de las distribuciones de posibilidad $A(x)$ y $A(y)$.

- Si $t_1 = t_2 = \ldots = t_n$, entonces el sistema evalúa la proposición sobre los datos dados por las distribuciones de posibilidad $A(x)$ y $A(y)$ como *cierta*, y le asocia un par de verdad $< T, t1 >$.
- Si no es así, entonces el sistema evalúa la proposición sobre los datos dados por las distribuciones de posibilidad $A(x)$ y $A(y)$ como *posible*, y le asocia un par de verdad $< P, max(t_1, t_2, \ldots, t_n) >$.

La condición que se establece en una consulta, puede estar compuesta de varias condiciones atómicas. Cuando la condición es compuesta, se evalúa mediante los conectivos conjunción, disyunción y negación que se definen de la siguiente forma:

a) Conjunción:

$$< c_1, t_1 > \wedge < c_2, t_2 > = < min(c_1, c_2), min(t_1, t_2) >$$

donde $min(T, T) = T, min(T, P) = Pymin(P, P) = P$

b) Disyunción:

1. Si $c_1 = c_2$ tenemos $< c_1, t_1 > \vee < c_2, t_2 > = < c_1, max(t_1, t_2) >$
2. Si $c_1 \neq c_2$, $c_1 = T$ y $c_2 = P$ tenemos:
   - Si $t_1 \geq t_2$ ó $t_1 < t_2$ y $t_1 \geq 0.5$ entonces $< T, t_1 > \vee < P, t_2 > = < T, t_1 >$
   - Si no, $< T, t_1 > \vee < P, t_2 > = < P, t_2 >$

c) Negación:

$$\neg < T, t > = < T, 1 - t >$$

$$\neg < P, t > = < P, 1 >$$

– Modelo Prade-Testemale

La estructura de datos sobre la que opera este modelo es básicamente la misma que la adoptada por el modelo anterior pero empleando medidas de posibilidad y necesidad para la satisfacción de las condiciones establecidas en la consulta [100].

Los dominios que se permiten son los siguientes:

1. Un conjunto finito de escalares. Por ejemplo: D={pelirrojo,castaño,negro,rubio}
2. Un conjunto finito de números. Por ejemplo: D={12,13,14}
3. Un conjunto de números difusos o etiquetas lingísticas. Por ejemplo: D={alto,medio,bajo,muy bajo}

Los valores permitidos para los dominios son:

1. Un único valor $d \in D$ perfectamente conocido. Por ejemplo, Altura(Pedro)=1.8 ó Color-Pelo(Pedro)=castaño. No se considera el caso de varios valores perfectamente conocidos (atributos multivaluados), como por ejemplo: Idiomas-que-habla (Pedro) = Inglés y Español.
2. Valor nulo, que incluye el valor desconocido y el no aplicable.
3. Una distribución de posibilidad sobre el dominio del atributo. Por ejemplo, Altura(Pedro)="no muy alto" donde no muy alto es un difuso sobre el dominio del atributo altura de la forma:

$$\{0.8/1.60, 0.9/1.65, 1/1.70, 1/1.75, 0.8/1.80\}$$

Llamaremos *relación básica* a aquella relación que se encuentra físicamente almacenada en la base de datos, es decir, que se corresponde con las relaciones usuales salvo que se admiten valores de dominio difusos. Llamaremos *relaciones derivadas* a las que se derivan de las básicas en el transcurso de la evaluación de una consulta.

Consideremos la tabla 2.6, en la que se almacenan datos de estudiantes junto con su calificacion en una asignatura:

donde $[21, 24]$ y $[8, 10]$ representan intervalos de valores y *"joven"*, *"muy joven"*, *"no muy mal"*,... son distribuciones de posibilidad previamente definidas. El concepto de "desconocido" es el mismo que el del modelo de Umano y el de "no aplicable" el mismo que el de "indefinido".

Para resolver una consulta del tipo *"Encontrar aquellos alumnos cuya calificación en programación sea mucho mayor que buena"*, tendríamos que partir de la definición

Nombre	**Edad**	**Nota**
María	joven	[8,10]
Juan	aprox. 24	Undefined
Luis	[21,24]	no muy mal
Laura	19	9.5
Olga	Unknown	bien
Félix	muy joven	aprox. 5
Julia	22	7

Tabla 2.6. Tabla ALUMNOS

de buena (mediante su distribución de posibilidad) y de una definición del comparador difuso mucho mayor que, como por ejemplo:

$$buena = \{0.3/7, 0.6/8, 0.9/9, 1/10\}$$

$$\mu_{>>}(u,v) = \begin{cases} 0 & u - v \leq 2 \\ 0.5 & u - v = 3 \\ 1 & u - v \geq 4 \end{cases}$$

El modelo resuelve primero la composición del comparador difuso mucho mayor que con la distribución de posibilidad dada para buena. Esto se realiza utilizando la regla de composición dada por Zadeh en [128]. Según esta regla, la composición de un subconjunto difuso A dado en el universo de discurso U y un operador relacional difuso R sobre $U \times V$ viene dada por:

$$A \circ R = max_{u \in U}(\mu_A(u) \wedge \mu_R(u,v))/v \in V$$

Tras componerse ambas funciones, se recuperarían las tuplas que satisfacen la consulta agrupadas en dos conjuntos según que la medida de compatibilidad elegida sea la posibilidad o la necesidad, esto es:

$$\Pi(F/A) = Sup_u\{\mu_F(u) \wedge \pi_A(u)\}$$

ó bien

$$N(F/A) = inf_u\{\mu_F(u) \vee (1 - \pi_A(u))\}$$

La condición impuesta a las tuplas a recuperar puede ser atómica o compuesta. En el caso de ser compuesta, los conectivos AND, OR y NOT vienen dados por la T-norma del mínimo, la T-conorma del máximo y por el complemento a 1 de la función de pertenencia, respectivamente.

El modelo distingue entre dos tipos de condiciones atómicas, $A\gamma a$ y $A\gamma B$, donde A y B representan los atributos a comparar, a es una constante y γ es un comparador que viene dado por una función de pertenencia μ, definida sobre el producto cartesiano de dos dominios, y que toma valores en el intervalo unidad.

– Modelo Zemankova-Kandel

Este modelo, ([138], [139]) se compone de tres partes:

a) Una base de datos de valores

b) Una base de datos explicativa donde se almacenan las definiciones de los conjuntos difusos utilizados

c) Un conjunto de reglas de traducción para el manejo de adjetivos y modificadores.

El planteamiento de consultas se hace de forma similar al modelo Prade-Testemale, salvo que la medidad de posibilidad que se emplea para encontrar la compatibilidad del subconjunto difuso F de la condición, con el valor del atributo A, viene dada por:

$$\mathcal{P}_A(F) = sup_u\{\mu_F(u) * \pi_A(u)\}$$

y la de certeza,

$$\mathcal{C}_A(F) = max\{0, inf\{\mu_F(u) * \pi_A(u)\}\}$$

donde esta última, es usada en lugar de la necesidad del modelo Prade-Testemale.

Al imponer condiciones a la hora de realizar una selección, se parte de una relación de similaridad definida sobre $D \times D$, a partir de la cual se construye cualquier otra relación de comparación.

◇ Conclusiones

Los modelos analizados presentan una serie de propiedades deseables a la hora de encontrar un modelo de bases de datos relacional difuso. De los posibilísticos, cabe destacar que la forma de representar la información difusa por medio de distribuciones de posibilidad parece ser muy adecuada para captar el significado de una amplia variedad de datos, si bien existe cierto tipo de información difusa que resulta más natural representar y comparar recurriendo al empleo de relaciones de similitud o de semejanza. En los capítulos siguientes, partiremos de un modelo generalizado [83] que recoja las alternativas más ventajosas de cada uno de los modelos expuestos, de manera que resulte lo más flexible y eficiente posible.

Capítulo 3

Aproximación Lógica de las Bases de Datos Difusas

Como hemos visto en el capítulo 2, existen dos grandes modelos o aproximaciones de Bases de Datos Difusas (BDD) en la literatura: El modelo de Relaciones de Similitud y el modelo Posibilístico.

Las ideas básicas de estos dos grandes modelos, junto con las nuevas perspectivas sobre el uso de la Lógica para representar y manipular Bases de Datos [102] [58], nos han llevado a establecer una nueva definición de Base de Datos Difusa desde el punto de vista de la Lógica, siguiendo los pasos marcados por los trabajos anteriormente mencionados para el caso clásico, y sin apartarnos del marco de la lógica de predicados de primer orden.

El objetivo primordial de este capítulo, es dar una definición lógica de BD difusa para aprovechar la versatilidad, flexibilidad y modularidad de la lógica de primer orden sin tener la enorme restricción de la versión clásica para BD, con respecto a la precisión de los datos. Junto con estas dos ventajas, nos colocamos, además, en el punto de partida de la definición del concepto de Base de Datos Difusa Deductiva.

Desde el punto de vista computacional, las ventajas son también evidentes: al no salirnos del marco de la lóica de primer orden, nos encontramos con la gran ventaja de que existen muchas herramientas y lenguajes de programación lógica cuya efectividad está sobradamente comprobada y que nos permitirían una implementación muy directa.

Por último, señalar que las ideas que vamos a desarrollar en el capítulo, ya se apuntaban en [121] y algunas de ellas aparecen ya publicadas en [120], [122], [96].

3.1 Definición Lógica de Base de Datos Difusa

Al igual que sucede en el caso *"no difuso"*, una Base de Datos Difusa puede verse, bien como una teoría especial de primer orden, o bien como una interpretación de un determinado lenguaje de primer orden. En primer lugar, definiremos una Base de Datos Difusa (BDD) como una interpretación de un lenguaje de primer orden, teniendo en cuenta conceptos previamente definidos en [102].

3.1.1 Definiciones y Conceptos

Consideraremos la existencia de un lenguaje relacional $L = (A, W)$ (tal y como lo ha definido Reiter), esto es:

- A es un alfabeto con un número finito de símbolos de constante y de predicado y sin símbolos de función.
- W es el conjunto de fbf que pueden obtenerse de A.
- Existe un conjunto no vacío de predicados unarios llamados *tipos*, que sirven para establecer los dominios de los atributos.
- Entre los predicados, existe un predicado binario especial, el $=$.

Con la idea de poder representar todos los elementos adicionales que aparecen en una BDD, será necesario introducir una serie de nuevos elementos a nuestro lenguaje relacional, que son los que definimos a continuación.

Definición 3.1 .

El **tipo MD.** *Impondremos, en primer lugar, que el alfabeto A contenga un símbolo de predicado unario adicional MD (Membership Degree), que servirá para modelizar*

el dominio de los grados de pertenencia para cualquier interpretación de L. Junto con el predicado MD, se incluirá también el símbolo de predicado binario $\geq$.

Definición 3.2 .

Sea τ *un símbolo de tipo del alfabeto A* ($\tau \in A$). *Diremos que* τ *es un* **tipo difuso** *si, y sólo si:*

- *Existe un símbolo de tipo simple* $NOP_\tau \in A$, *que nos permitirá representar los nombres de las distribuciones de posibilidad que se definan sobre el dominio de* τ.
- *Existe un símbolo de predicado ternario* $POS_\tau \in A$, *que se utilizará para representar las distribuciones de posibilidad que se definan sobre el dominio de* τ.
- *Existe un símbolo de predicado ternario* $\cong_\tau \in A$, *para representar la relación de* igualdad *entre las distribuciones de posibilidad.*

En base a esta definición, extendemos los conceptos de Lenguaje Relacional e Interpretación Relacional al caso difuso, como sigue:

Definición 3.3 .

Sea L un lenguaje $L = (A, W)$. *Diremos que L es un* **Lenguaje Relacional Difuso (LRD)** *si, y sólo si:*

- *Incluye al tipo MD entre sus símbolos.*
- *Incluye, al menos, un símbolo de tipo difuso.*

3.2 BDD como Interpretación Relacional Difusa

En esta primera parte, definiremos una BDD como una Interpretación Relacional Difusa, usando el concepto de LRD. Dado que no será válida cualquier interpretación, será necesario incluir, a su vez, un conjunto de reglas de integridad a verificar por la BDD, de manera que se preserve la semántica del tipo MD así como la de los tipos difusos.

Definición 3.4 .

Sea L un LRD y sea $I = (U, a)$ una interpretación de dominio U y función de asignación a. Diremos que I es una **Interpretación Relacional Difusa (IRD)** *si, y sólo si:*

- *I es una interpretación relacional en el sentido de Reiter.*
- *Existe un conjunto finito $T_v \subset [0,1]$ tal que:*
  - $0 \in T_v$
  - $1 \in T_v$
  - $T_v \subset U$
  - $a(MD) = T_v$
- *I es un modelo para el siguiente conjunto de reglas de integridad:*
  - Reglas para establecer la semántica del tipo MD.

    R1. $\forall(x,y)\ (\geq(x,y) \longrightarrow MD(x) \wedge MD(y))$

    R2. $\forall(x)\ (MD(x) \longrightarrow \geq(x,x))$

    R3. $\forall(x,y)\ (\geq(x,y) \wedge \geq(y,x) \longrightarrow =(x,y))$

    R4. $\forall(x,y,z)\ \ (\geq(x,y) \wedge \geq(y,z) \longrightarrow \geq(x,z))$

    Como puede verse, este conjunto de reglas (que se corresponden con las propiedades Reflexiva, Simétrica y Transitiva) establece que $\geq$ es una relación de orden sobre el tipo MD.

  - Reglas para establecer la semántica de los tipos difusos.

    Para cada tipo difuso τ de L deben verificarse las siguientes reglas:

    R5. $\forall(x,y,z)\ (POS_\tau(x,y,z) \longrightarrow NOP_\tau(x) \wedge \tau(y) \wedge MD(z) \wedge \neg(=(z,0)))$.

Esta regla define el dominio del predicado POS_τ, esto es, los tipos de sus argumentos y las restricciones sobre los mismos.

R6. $\forall(x)\ (NOP_\tau(x) \longrightarrow (=(x, Und_\tau)) \vee (\exists(y,z)\ (POS_\tau(x,y,z)))$.
Esta regla obliga a que cada nombre de distribución tenga su correspondiente definición en términos de la distribución de posibilidad que lleva asociado, exceptuando los nombres de los valores indefinidos, Und_τ, que no llevarán asociados el correspondiente predicado POS_τ.

R7. $\forall(x)\ (\tau(x) \longrightarrow POS_\tau(x,x,1) \wedge (\neg\exists(y)\ (POS_\tau(x,y,1) \wedge \neg =(x,y))))$.
Esta regla incluye la extensión del predicado τ dentro de la extensión del predicado POS_τ tratándolos como conjuntos unarios.

* Reglas para establecer la semántica del operador de igualdad difuso (aproximadamente igual).

R8. $\forall(x,y,z)\ (\cong_\tau (x,y,z) \longrightarrow NOP_\tau(x) \wedge NOP_\tau(y) \wedge MD(z))$.
Esta regla establece los tipos de los argumentos del predicado $\cong_\tau$.

R9. $\forall(x,y,z,w)\ (\cong_\tau (x,y,z) \wedge \geq(z,w) \longrightarrow \cong_\tau (x,y,w))$.
Esta regla establece que si se da la igualdad entre las distribuciones x e y en grado z, también se da en grado inferior a z.

R10. $\forall(x,y),\ \cong_\tau (x,y,z) \wedge =(x,y) \longrightarrow =(z,1)$

R11. $\forall(x,y,z)\ (\cong_\tau (x,y,z) \longrightarrow \cong_\tau (y,x,z))$.
Esta regla establece la simetría del predicado "aproximadamente igual". *Si, adicionalmente, se impone la transitividad, la relación $\cong$ es una relación de similitud.*

Nota: En adelante, y con el objeto de simplificar la notación, identificaremos los valores del dominio U con sus correspondiente símbolos en A.

Definición 3.5 .

Siguiendo la teoría general dada en [102], y teniendo en cuenta las nuevas definiciones de LRD e IRD, definimos una **Base de Datos Relacional Difusa (BDRD)** *como un triple* (L, I, RI) *donde:*

- L *es un LRD.*
- I *es una IRD.*
- RI *es un conjunto de restricciones de integridad específico, que incluye para cada predicado* P *n-ario de* L*, la siguiente regla:*

$$\forall(x_1, x_2, \ldots, x_n)\ (P(x_1, x_2, \ldots, x_n) \longrightarrow \tau_1(x_1) \wedge \tau_2(x_2) \wedge \ldots \wedge \tau_n(x_n))$$

donde los τ_i *son tipos simples.*

3.2.1 Definición de un Lenguaje de Consulta

Dado que las definiciones dadas en la sección anterior caracterizan una BDRD como una estructura lógica de primer orden, el lenguaje de consulta asociado podrá expresarse por medio del Cálculo de Predicados de primer orden como sigue:

Consideremos un LRD $L = (A, W)$. LLamaremos **consulta** a cualquier expresión de la forma:

$$< X/\Phi \ \mid \ W(X) >$$

donde X/Φ representa la secuencia $x_1/\tau_1, \ldots, x_n/\tau_n$, x_i son variables de A y τ_i son tipos simples y donde $W(X) \in W$ es una fbf del lenguaje L en la que $x_1, \ldots, x_n$ son variables libres.

De la misma forma que para el caso *no difuso*, se define una respuesta a una consulta como sigue. Diremos que la tupla $C =< c_1, c_2, \ldots, c_n >$, donde las c_i son constantes, es una **respuesta** a la consulta $Q =< X/\Phi \ \mid \ W(X) >$, hecha sobre una BDRD $B = (L, I, RI)$, si, y sólo si:

Nombre	**Edad**	**Curso**	**Comportamiento**
Jose	Muy Joven	1	Regular
Maria	14	1	Bueno
Antonio	Joven	2	Muy Bueno
Carlos	16	3	Malo
Susana	Media	5	Ejemplar

Tabla 3.1. Tabla ALUMNOS

– $\forall j,\ j \in \{1, \ldots, n\}$ la expresión $\tau_j(c_j)$ es cierta en I.

– $W(C)$ es cierta en I.

3.2.2 Ejemplo

Supongamos la tabla 3.1 de una instancia de BDD, en la que aparecen, como atributos no difusos (*crisp*) el **Nombre** y el **Curso**. Como atributo difuso, aparece la **Edad**, mientras que el **Comportamiento**, aún teniendo una representación crisp, estará sujeto a relaciones de similitud tipo Buckles-Petry.

∗ **Elementos del alfabeto A**:

– *Constantes*:

Jose, Maria, Antonio, Carlos, Susana, Muy-Joven, 14, Joven, 16, Media, 1, 2, 3, 5, Regular, Bueno, Muy-Bueno, Malo, Ejemplar.

– *Predicados*:

$Alumno(-,-,-,-)$, $Nombre(-)$, $Edad(-)$, $Curso(-)$, $Comp(-)$, $MD(-)$, $NOM_{Edad}(-)$, $NOM_{Comp}(-)$, $POS_{Edad}(-,-,-)$, $POS_{Comp}(-,-,-)$, $\cong_{Edad}(-,-,-)$, $\cong_{Comp}(-,-,-)$, $=(-,-)$, $\geq(-,-)$.

– *Tipos Difusos*:

Edad: Sus predicados asociados son NOM_{Edad} y POS_{Edad}.

Comp: Sus predicados asociados son NOM_{Comp}, POS_{Comp}.

- *Tipos*:

  Son todos los predicados unarios del conjunto anterior.

- *Símbolos Adicionales*:

  Todos los conectivos y separadores necesarios para construir las fórmulas.

* **Interpretación Difusa**

- *Dominio*:

  El conjunto de constantes del alfabeto A.

- *Extensiones de los Predicados*:

  Extensión del predicado ALUMNO.

  Alumno(Jose,Muy Joven,1,Regular)
  Alumno(Maria,14,1,Bueno)
  Alumno(Antonio,Joven,2,Muy Bueno)
  Alumno(Carlos,16,3,Malo)
  Alumno(Susana,Media,5,Ejemplar)

  Extensión del predicado NOMBRE.

  Nombre(Jose)
  Nombre(Maria)
  Nombre(Antonio)
  Nombre(Carlos)
  Nombre(Susana)

  Extensión del predicado EDAD.

  Edad(Muy Joven)
  Edad(14)
  Edad(Joven)
  Edad(16)
  Edad(Media)

  Extensión del prdicado CURSO.

Curso(1)
Curso(2)
Curso(3)
Curso(5)

Extensión del predicado COMPORTAMIENTO.

Comp(Regular)
Comp(Bueno)
Comp(Muy Bueno)
Comp(Malo)
Comp(Ejemplar)

Extensión del predicado MD.

MD(0.0)
MD(0.1)
MD(0.2)
...
MD(1.0)

Extensión del predicado $\geq$.

$\geq$(0.0,0.0)
$\geq$(0.1,0.0)
$\geq$ (0.1,0.1)
$\geq$(2.0,0.0)
...
$\geq$(1.0,0.9)
$\geq$(1.0,1.0)

Extensión del predicado =.

=(Jose,Jose)
=(Maria,Maria)
=(Antonio,Antonio)
...
=(Ejemplar,Ejemplar)

Extensión del predicado NOM para los atributos difusos.

NOM_{Edad}(14)
NOM_{Edad}(15)
...
NOM_{Edad}(Joven)
NOM_{Edad}(Muy Joven)
NOM_{Edad}(Media)
NOM_{Edad}(Mayor)
NOM_{Edad}(Muy Mayor)

NOM_{Comp}(Muy Malo)
NOM_{Comp}(Malo)
NOM_{Comp}(Regular)
NOM_{Comp}(Bueno)
NOM_{Comp}(Muy Bueno)
NOM_{Comp}(Ejemplar)

Extensión del predicado POS para los atributos difusos.

POS_{Edad}(14,14,1.0)
POS_{Edad}(15,15,1.0)
...
POS_{Edad}(40,40,1.0)
POS_{Edad}(Muy Joven,14,1)
POS_{Edad}(Muy Joven,15,0.9)
POS_{Edad}(Joven,14,0.6)
POS_{Edad}(Joven,15,1)
POS_{Edad}(Joven,16,0.8)
POS_{Edad}(Joven,17,0.6)
POS_{Edad}(Joven,18,0.5)
POS_{Edad}(Joven,19,0.4)

POS_{Edad}(Media,19,0.5)
POS_{Edad}(Media,20,0.7)
POS_{Edad}(Media,21,0.8)
POS_{Edad}(Media,22,0.9)
POS_{Edad}(Media,23,1)
POS_{Edad}(Media,24,1)
POS_{Edad}(Media,25,1)
POS_{Edad}(Media,26,0.8)
POS_{Edad}(Media,27,0.5)
POS_{Edad}(Media,28,0.4)
...

así sucesivamente para el resto de las etiquetas definidas sobre la edad.

POS_{Comp}(Muy Malo,Muy Malo,1)
POS_{Comp}(Regular,Regular,1)
POS_{Comp}(Muy Bueno,Muy Bueno,1)

POS_{Comp}(Malo,Malo,1)
POS_{Comp}(Bueno,Bueno,1)
POS_{Comp}(Ejemplar,Ejemplar,1)

Extensión del predicado $\cong$ para los atributos difusos.

$\cong_{Edad}$(14,14,1.0)
$\cong_{Edad}$(14,15,0.9)
...
$\cong_{Edad}$(15,15,1.0)
$\cong_{Edad}$(15,16,0.9)
...
$\cong_{Edad}$(Muy Joven,Joven,0.6)
$\cong_{Edad}$(Joven,Media,0.8)
$\cong_{Edad}$(Media,Mayor,0.5)
$\cong_{Edad}$(Mayor,Muy Mayor,0.4)
$\cong_{Edad}$(Joven,Joven,1)
...
$\cong_{Edad}$(Muy Mayor,Muy Mayor,1)

más las tuplas que hacen la relacion simétrica.

$\cong_{Comp}$(Muy Malo,Muy Malo,1)
...
$\cong_{Comp}$(Ejemplar,Ejemplar,1)
$\cong_{Comp}$(Muy Malo,Malo,0.8)
$\cong_{Comp}$(Muy Malo,Regular,0.5)
$\cong_{Comp}$(Malo,Regular,0.7)
$\cong_{Comp}$(Malo,Bueno,0.4)
$\cong_{Comp}$(Regular,Bueno,0.6)
$\cong_{Comp}$(Regular,Muy Bueno,0.5)
$\cong_{Comp}$(Bueno,Muy Bueno,0.7)
$\cong_{Comp}$(Bueno,Ejemplar,0.6)
$\cong_{Comp}$(Muy Bueno,Ejemplar,0.9)
...

más las tuplas que hacen la relacion simétrica.

– Reglas de Integridad

∗ *Reglas para el tipo MD*

I1. $\forall\ (x,y),\ (\geq (x,y) \longrightarrow MD(x) \wedge MD(y))$

I2. $\forall x,\ (MD(x) \longrightarrow \geq (x,x))$

I3. $\forall\ (x,y),\ \geq (x,y) \wedge \geq (y,x) \longrightarrow = (x,y))$

I4. $\forall\ (x,y,z),\ (\geq (x,y) \wedge \geq (y,z) \longrightarrow \geq (x,z))$

∗ *Reglas para el tipo difuso EDAD*

I5. $\forall\ (x,y,z),\ POS_{Edad}(x,y,z) \longrightarrow NOM_{Edad}(x) \wedge Edad(y) \wedge MD(z) \wedge \neg = (z,0))$

I6.$\forall x,\ POS_{Edad}(x) \longrightarrow \exists\ (y,z),\ POS_{Edad}(x,y,z))$

I7. $\forall x,\ Edad(x) \longrightarrow POS_{Edad}(x,x,1) \wedge \neg\exists\ (y,z),\ POS_{Edad}(x,y,z) \wedge \neg = (x,y))$

I8. $\forall\ (x,y,z),\ (\cong_{Edad}\ (x,y,z) \longrightarrow NOM_{Edad}(x) \wedge NOM_{Edad}(y) \wedge MD(z))$

I9. $\forall\ (x,y,z),\ \cong_{Edad}\ (x,y,z) \longrightarrow \cong_{Edad}\ (y,x,z))$

∗ *Reglas para el tipo difuso COMP*

I10. $\forall\ (x,y,z),\ POS_{Comp}(x,y,z) \longrightarrow NOM_{Comp}(x) \wedge COMP(y) \wedge MD(z) \wedge \neg = (z,0))$

I11.$\forall x,\ POS_{Comp}(x) \longrightarrow \exists\ (y,z),\ POS_{Comp}(x,y,z))$

I12.$\forall x,\ Comp(x) \longrightarrow POS_{Comp}(x,x,1) \wedge \neg\exists\ (y,z),\ POS_{Comp}(x,y,z) \wedge \neg = (x,y))$

I13. $\forall\ (x,y,z),\ (\cong_{Comp}\ (x,y,z) \longrightarrow NOM_{Comp}(x) \wedge NOM_{Comp}(y) \wedge MD(z))$

I14. $\forall\ (x,y,z),\ \cong_{Comp}\ (x,y,z) \longrightarrow \cong_{Comp}\ (y,x,z))$

– Reglas específicas

I15. $\forall\ (x,y,z),\ (POS_{Comp}(x,y,z) \longrightarrow = (x,1.0))$

Que impone que el atributo COMPORTAMIENTO tome sólo como valores conjuntos clásicos.

I16. $\forall\,(x,y,z,t),\ (ALUMNO(x,y,z,t) \longrightarrow Nombre(x) \wedge Edad(y) \wedge Curso(z) \wedge Comp(t))$

Que establece el esquema de la tabla ALUMNOS.

Todas estas ideas quedan recogidas y desarrolladas en [120].

3.3 Representación Lógica de los Modelos de BDD más Representativos

Veamos ahora de qué manera, los dos grandes modelos de BDD que aparecen en la literatura, el de Relaciones de Similitud [12],[13]* y el Posibilístico [138] [51] [76] [99] [118]..., en sus distintas vertientes, pueden quedar representados desde el punto de vista lógico, gracias a la nueva definición lógica de BDD realizada.

3.3.1 Representación del Modelo de Relaciones de Similitud

Para identificar el Modelo de Relaciones de Similitud (MRS) dentro de nuestra definición lógica de BDD, nos centraremos en dos puntos, la estructura de datos y el cálculo de dominios.

◇ Identificación de la Estructura de Datos

Recordemos que, en el MRS, la estructura de los datos está compuesta por un conjunto de relaciones no normalizadas en las que los valores de los atributos pueden ser conjuntos. Además, existe una relación de similitud sobre cada uno de los dominios básicos existentes.

*También llamado modelo Buckles-Petry debido a sus principales autores.

Sea $B = (L, I, RI)$ una BDD tal como la definimos en def. 3.5. B será una BDD según el MRS, si verifica las siguientes condiciones:

* *Condición de Atributos Difusos*. Cualquier tipo $\tau \in L$, $\tau \neq MD$ es un tipo difuso. Esta condición establece que cualquier atributo de la BD debe ser difuso, lo cual es coherente con el hecho de que el MRS no impone la existencia de ningún atributo preciso a la hora de definir las relaciones. Hay que hacer notar que esta condición junto con las reglas que establecen la semántica del operador $\cong$ (reglas R8 y R9), nos permiten asegurar que existe una relación de similitud asociada a cada dominio.

* *Condición de Conjuntos Precisos*. Para cualquier tipo $\tau \in L$ debe verificarse:

$$\forall\ (x, y, z)\ (POS_\tau(x, y, z) \longrightarrow = (z, 1)$$

que establece que los valores de los atributos deben ser conjuntos clásicos.

* *Definición de la Relación de Similitud*. Para cualquier tipo $\tau \in L$ deben verificarse:

1. $\forall\ (x, y, z)\ (\cong_\tau (x, y, z) \wedge = (z, 1) \longleftrightarrow = (x, y))$
   que establece que una relación de similitud con grado 1 equivale a la igualdad, y
2. $\forall\ (x, y, z)\ (\cong_\tau (x, y, z) \longleftrightarrow (\tau(y) \wedge (\forall w\ (POS_\tau(x, w, 1) \longrightarrow \exists v\ (\cong_\tau (y, w, v) \wedge \geq (v, z)) \vee \tau(x) \wedge (\forall w\ (POS_\tau(y, w, 1) \longrightarrow \exists v\ (\cong_\tau (x, w, v) \wedge \geq (v, z))))$
   que establece que al menos uno de los dos valores a comparar debe ser un conjunto unitario. El nivel de similaridad entre un valor (conjunto unitario) x y un conjunto Y vendrá dado por:

$$z = Min_{w \in Y}\{S(x, w)\}$$

donde S es la Relación de Similitud asociada al tipo τ ($\cong_\tau$).

◇ Identificación del Lenguaje de Consulta

Analizando el cálculo orientado a dominios propuesto en [14] para el MRS, puede verse que éste no difiere en esencia del propuesto en nuestra definición (secc. 3.2.1), de manera que será suficiente con mostrar cómo se construirán internamente las fórmulas. Dado que las fórmulas en el MRS son secuencias de átomos conectados entre sí por los conectivos lógicos clásicos, sólo tendremos que preocuparnos por representar dichos átomos en nuestro lenguaje. El MRS trabaja con dos tipos de átomos:

- $x\Theta y$ *con nivel* α, que representa una operación de comparación difusa entre los elementos de un dominio τ. Un par de valores a y b hacen verdadero este átomo, si el grado asociado a la comparación $a\Theta b$ es mayor o igual a α. En nuestro lenguaje, Θ vendrá representado por un predicado ternario asociado al tipo τ, que verifique, al menos, la siguiente regla:

$$\forall\ (x, y, z)\ (\Theta(x, y, z) \longrightarrow (\tau(x) \wedge \tau(y) \wedge MD(z))$$

  con lo que el átomo anterior quedará representado por la expresión:

$$\exists x\ (\Theta(x, y, z) \wedge \geq (z, \alpha))$$

- $R(y_1, y_2, \ldots, y_n)$ *con nivel* $(\alpha_1, \alpha_2, \ldots, \alpha_m)$, donde R es un predicado n-ario, $y_j,\ j \in \{1, 2, \ldots, m\}$ son variables y $y_j,\ j \in \{m+1, m+2, \ldots, n\}$ son constantes, de dominios respectivos τ_j. Este átomo así construido, quedará traducido de la siguiente forma:

$$\exists\ (v_1, v_2, \ldots, v_n)\ (R(w_1, w_2, \ldots, w_m, y_{m+1}, y_{m+2}, \ldots, y_n) \wedge (\cong_{\tau_1}\ (w_1, y_1, v_1) \wedge$$
$$\geq (v_1, \alpha_1)) \wedge \ldots \wedge (\cong_{\tau_m}\ (w_m, y_m, v_m) \wedge \geq (v_m, \alpha_m)$$

Así pues, podemos asegurar que cualquier consulta en el cálculo propuesto para el MRS tiene su correspondiente representación en nuestro modelo. Más aún, dado que en [14] se demuestra que dicho cálculo es equivalente al álgebra relacional para su modelo, queda automáticamente demostrado que ésta lo es también respecto del cálculo de nuestro modelo.

3.3.2 Representación del Modelo Posibilístico

La principal característica de los Modelos Posibilísticos (MP) es que cada atributo puede tomar cualquier valor difuso, lo cual viene expresado por una distribución de posibilidad. Aunque parezca sencillo trascribir un modelo de este tipo a nuestro lenguaje, existen algunas complicaciones originadas, principalmente, por la forma en que se realizan las consultas.

Tanto en el modelo de Zemankova-Kandel [138] como en el de Prade-Testemale [99], aparecen una serie de características comunes en lo que a dichas consultas se refiere, y que son las siguientes:

* Cualquier condición atómica de la forma $x\Theta y$, donde x e y son valores difusos, implica la existencia de una relación difusa equivalente, aplicable a valores precisos (*"crisp"*). Esta característica debe verificarla también el predicado $\cong$.

* Cualquier condición atómica lleva asociada dos grados de cumplimiento, que se corresponden con la *posibilidad* y la *necesidad*, respectivamente. Esto nos obligará, no sólo a manejar dos predicados asociados a cada comparador difuso, sino también a introducir operaciones adicionales, tales como suma, producto, máximo,... definidos sobre los grados de cumplimiento, para poder operar con ellos.

◇ Identificación de la Estructura de Datos

Para llevar a cabo esta identificación, habrá que tener presentes las dos puntualizaciones anteriores, esto es, se hace necesario introducir nuevas operaciones en relación con el tipo MD, y nuevos predicados en relación con las medidas de posibilidad y necesidad que se van a manejar.

Sea $B = (L, I, RI)$ una BDD. Diremos que B es una BDD según el MP, si verifica las siguientes condiciones:

* *Condición de Atributos Difusos*. Tiene el mismo sentido que para el MRS.

* *Definición de Operaciones sobre el Dominio de MD*. Hay que incluir todas las reglas de definición de los operadores máximo, mínimo, suma y producto en relación con el dominio del tipo MD, como sigue:

  1. *Definición del Mínimo y el Máximo*: Existen dos predicados MAX y MIN asoicados al tipo MD que verifican:
     - **E1.**- $\forall(x,y),\ (MD(x) \wedge MD(y) \longrightarrow \exists z,\ (MD(z) \wedge MIN(x,y,z)))$
     - **E2.**- $\forall(x,y,z),\ (MIN(x,y,z) \longleftrightarrow (MD(x) \wedge MD(y) \wedge MD(z) \wedge ((\geq(x,y) \wedge = (y,z)) \vee (\geq(y,x) \wedge = (x,z)))))$
     - **E3.**- $\forall(x,y),\ (MD(x) \wedge MD(y) \longrightarrow \exists MD(z) \wedge MAX(x,y,z))$
     - **E4.**- $\forall(x,y,z),\ (MAX(x,y,z)) \longleftrightarrow (MD(x) \wedge MD(y) \wedge MD(z) \wedge (\geq(x,y) \wedge = (y,z)) \vee (\geq(y,x) \wedge = (x,z))))$
  2. *Definición de la Adición y el Complemento*: Existen dos predicados + y NEG que deben verificar:
     - **E5.**- $\forall(x,y,z),\ (+(x,y,z) \longrightarrow MD(x) \wedge MD(y) \wedge MD(z))$
     - **E6.**- $\forall(x,y,z,v_1,v_2,w_1,w_2),\ (+(x,y,v_1) \wedge +(v_1,z,w_1) \wedge +(y,z,v_2) \wedge +(x,v_2,w_2) \longrightarrow = (w_1,w_2))$
     - **E7.**- $\forall(x,y,z),\ (+(x,y,z) \longrightarrow +(y,x,z))$
     - **E8.**- $\forall(x,y,z)\ (+(x,y,z) \longrightarrow \geq(z,x) \wedge \geq(z,y))$
     - **E9.**- $\forall x,\ (MD(x) \longrightarrow +(x,0,x))$
     - **E10.**- $\forall x,\ (MD(x) \longrightarrow \exists y,\ (MD(y) \wedge NEG(x,y)))$
     - **E11.**- $\forall(x,y),\ (NEG(x,y) \longleftrightarrow +(x,y,1))$
  3. *Definición del Producto*: Existe un predicado $*$ verificando:
     - **E12.**- $\forall(x,y),\ (MD(x) \wedge MD(y) \longrightarrow \exists z,\ (MD(z) \wedge *(x,y,z)))$
     - **E13.**- $\forall(x,y,z,v_1,v_2,w_1,w_2),\ (*(x,y,v_1) \wedge *(v_1,z,w_1) \wedge *(y,z,v_2) \wedge *(x,v_2,w_2) \longrightarrow = (w_1,w_2))$
     - **E14.**- $\forall(x,y,z),\ (*(x,y,z) \longrightarrow *(y,x,z))$
     - **E15.**- $\forall x,\ (MD(x) \longrightarrow *(x,1,x))$
     - **E16.**- $\forall x,\ (MD(x) \longrightarrow *(x,0,0))$
     - **E17.**- $\forall(x,y,z),\ (*(x,y,z) \wedge \neg = (x,0) \wedge \neg = (y,0) \longrightarrow \geq(x,z) \wedge \geq(y,z))$

- **E18.-** $\forall(x,y,z,v_1,v_2,w_1,w_2),\ (+(x,y,v_1)\wedge *(v_1,z,w_1)\wedge *(x,z,v_2)\wedge *(y,z,v_3)\wedge +(v_2,v_3,w_2) \longrightarrow = (w_1,w_2))$

Estas reglas definen $*$ como una operación en $[0,1]$ con las mismas propiedades que el producto numérico clásico. Debe señalarse que que el producto numérico no puede definirse correctamente en un subconjunto finito de $[0,1]$, pero un proceso de redondeo resuelve este problema.

* *Condiciones para la Definición del Predicado* $\cong$. Se trata de añadir reglas asociadas al predicado $\cong_\tau$, asociado a cada tipo difuso, de manera que su comportamiento sea idéntico al correspondiente operador *"aproximadamente igual"* de los modelos Prade-Testemale y Zemankova-Kandel. Las reglas correspondientes serían:

1. *Definición de Relación Difusa Básica*: Para cada tipo difuso τ, existirá un predicado MU_τ asociado verificando:
   - **A1.-** $\forall(x,y),\ (\tau(x)\wedge\tau(y)\longrightarrow\exists z,\ (MD(z)\wedge MU_\tau(x,y,z))$
   - **A2.-** $\forall x,\ (\tau(x)\longrightarrow MU_\tau(x,x,1))$
   - **A3.-** $\forall(x,y,z),\ (MU_\tau(x,y,z)\longrightarrow MU_\tau(y,x,z))$

   Como puede verse, se han impuesto las condiciones mínimas para que MU_τ se comporte como una relación de similitud. MU_τ representa la relación *"aproximadamente igual"* definida sobre el dominio τ.
2. *Definición del Predicado asociado a los Grados de Posibilidad*: Para cada tipo τ, existe un predicado $\cong_{\pi_\tau}$ verificando:
   - **A4.-** $\forall(x,y,z),\ (\cong_{\pi_\tau}(x,y,z)\longrightarrow NOM_\tau(x)\wedge NOM_\tau(y)\wedge MD(x))$
   - **A5.-** $\forall(x,y),\ (NOM_\tau(x)\wedge NOM_\tau(y)\longrightarrow\exists z,\ (\cong_{\pi_\tau}(x,y,z)))$

   La siguiente regla se refiere al tratamiento concreto que cada modelo propone para el grado de posibilidad, de modo que dependerá del modelo.
   - **Modelo Prade-Testemale**

     Por simplicidad, definimos la siguiente fórmula bien formulada:

$$F(x,y,zr,xr,yr)\equiv\exists(r_1,r_2,r_3,r_4),$$

$$(POS_\tau(x,xr,r_1)\wedge POS_\tau(y,yr,r_2)\wedge$$

$$MU_\tau(xr, yr, r_3) \wedge MIN(r_1, r_2, r_4) \wedge MIN(r_3, r_4, zr))$$

cuyo signicado se corresponde con la expresión

$$zr = min(pos_x(xr), pos_y(yr), \mu(xr, yr))$$

de modo que la regla que define el predicado $\cong_{\pi_\tau}$ queda expresada como:
A6.-$\forall(x, y, z),\ (\cong_{\pi_\tau}(x, y, z) \longleftrightarrow exists(x_1, y_1),\ (F(x, y, z, x_1, y_1))$
$\wedge\forall(x_2, y_2, z_2),\ (F(x, y, z_2, x_2, y_2) \longrightarrow \geq ()z, z_2))$

- **Modelo Zemankova-Kandel**
  La fórmula bien formulada F será, en este caso:

$$F(x, y, zr, xr, yr) \equiv \exists(r_1, r_2, r_3, r_4),\ (POS_\tau(x, xr, r_1) \wedge POS_\tau(y, yr, r_2) \wedge$$

$$MU_\tau(xr, yr, r_3) \wedge MIN(r_1, r_3, r_4) \wedge *(r_2, r_4, zr))$$

cuyo significado se corresponde con la expresión

$$zr = min(pos_x(xr), \mu(xr, yr), pos_y(yr))$$

En este caso la regla **A6** permanece igual.

3. *Definición del Predicado asociado a los grados de Necesidad*: Para cada tipo difuso τ, se define un predicado $\cong_{N_\tau}$ verificando:
   - **A7.**- $\forall(x, y, z),\ (\cong_{N\tau}(x, y, z) \longrightarrow NOM_\tau(x) \wedge NOM_\tau(y) \wedge MD(z))$
   - **A8.**- $\forall(x, y),\ (NOM_\tau(x) \wedge NOM_\tau(y) \longrightarrow \exists z,\ (\cong_{N\tau}(x, y, z))$

   En este caso, será necesario realizar de nuevo la distinción entre los modelos Prade-Testemale y Zemankova-Kandel.

   - **Modelo Prade-Testemale**
     En este caso, F viene dada por:

$$F(x, y, zr, xr, yr) \equiv exists(r_1, r_2, r_3, r_4, r_5, r_6),$$

$$(POS_\tau(x, xr, r_1) \wedge POS_\tau(y, yr, r_2) \wedge$$

$$MU_\tau(xr, yr, r_3) \wedge NEG(r_1, r_4) \wedge NEG(r_2, r_5) \wedge$$

$$MAX(r_4, r_5, r_6) \wedge MAX(r_3, r_6, zr))$$

que se corresponde con la expresión

$$zr = max(1 - pos_x(xr), \mu(xr, yr), 1 - pos_y(yr))$$

y la regla que define el predicado será:
A9.- $\forall(x,y,z),\ (\cong_{\pi,N}(x,y,z) \longleftrightarrow \exists(x_1,y_1),(F(x,y,z,x_1,y_1)) \wedge$
$\forall(x_2,y_2,z_2),\ (F(x,y,z_2,x_2,y_2) \longrightarrow \geq(z_2,z)))$

- **Modelo Zemankova-Kandel**
  Ambos modelos son bastante diferentes en lo que a la medida inferior se refiere (llamada *"certeza"* en [138]), de manera que tanto la expresión F como la regla asociada son esencialmente distintas. En este caso, F tiene la forma:

$$F(x,yr,zr) \equiv \exists(r_1,r_2,r_3),\ (POS_\tau(x,r_1,r_2) \wedge$$

$$MU_\tau(r_1,yr,r_3) \wedge MIN(r_2,r_3,zr)) \wedge$$

$$\forall(r_4,r_5,r_6,z_1),\ (POS_\tau(x,r_4,r_5) \wedge$$

$$MU\tau(r_4,yr,r_6) \wedge MIN(r_2,r_3,z_1) \longrightarrow \geq(zr,z_1))$$

  que representa la expresión

$$zr = max_{x,r} min(pos_x(x,r), \mu(r,yr))$$

  siendo la regla asociada
  A10.-$\forall(x,y,z),\ (\cong_{N_\tau}(x,y,z) \longleftrightarrow \exists(yr,zr,vr),\ (F(x,yr,zr) \wedge$
  $POS_\tau(y,yr,vr) \wedge *(vr,zr,z)) \wedge \forall(ys,zs,vs,z_2),\ (F(x,ys,zs) \wedge$
  $POS_\tau(y,ys,vs) \wedge *(vs,zs,z_2) \longrightarrow \geq(z_2,z))$

Esta última regla completa la definición del comparador difuso *"aproximadamente igual"*, cuya existencia se ha impuesto para cada tipo difuso existente. Es obvio que pueden definirse otros comparadores difusos por medio de los predicados adecuados. Así pues, en lo sucesivo adoptaremos la notación Θ_{π_τ} y Θ_{N_τ} para las medidas superior e inferior, respectivamente, de los grados de cumplimiento de un comparador difuso Θ, definido para un tipo τ.

◇ Identificación del Lenguaje de Consulta

La identificación del lenguaje de consulta no resulta directa, dado que los MP manejan las relaciones por medio de un álgebra relacional.

Pasamos a describir brevemente cómo sería la correspondencia entre ambos lenguajes de consulta.

Las operaciones clásicas sobre conjuntos (unión, intersección, diferencia y producto cartesiano), tienen una trascripción casi directa, como sigue:

$$R \cup S \equiv < x_1, x_2, \ldots, x_n / \tau_1(x_1), \tau_2(x_2), \ldots, \tau_n(x_n) >$$

$$\mid \; R(x_1, x_2, \ldots, x_n) \vee S(x_1, x_2, \ldots, x_n)$$

$$R \cap S \equiv < x_1, x_2, \ldots, x_n / \tau_1(x_1), \tau_2(x_2), \ldots, \tau_n(x_n) > \mid$$

$$R(x_1, x_2, \ldots, x_n) \wedge S(x_1, x_2, \ldots, x_n)$$

$$R - S \equiv < x_1, x_2, \ldots, x_n / \tau_1(x_1), \tau_2(x_2), \ldots, \tau_n(x_n) > \mid$$

$$R(x_1, x_2, \ldots, x_n) \vee \neg S(x_1, x_2, \ldots, x_n)$$

$$R \times S \equiv < x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_m / \tau_1(x_1), \tau_2(x_2), \ldots, \tau_n(x_n),$$

$$\nu_1(y_1), \nu_2(y_2), \ldots, \nu_n(y_n) \;\mid\; R(x_1, x_2, \ldots, x_n) \wedge S(y_1, y_2, \ldots, y_m)$$

donde τ_i y ν_j son los respectivos conjuntos de dominios de R y S.

La operación de *selección* utilizando una medida de posibilidad quedaría expresada:

$$R \; donde \; x_i \Theta Y \equiv < x_1, x_2, \ldots, x_m / \tau_1(x_1), \tau_2(x_2), \ldots, \tau_n(x_m),$$

$$MD(v) \; / \; R(x_1, x_2, \ldots, x_m) \wedge \Theta_{\Pi_\tau}(x_i, Y, v) >$$

donde Θ_{Π_τ} es un comparador difuso asociado al tipo τ y Y es una constante difusa de τ. De manera análoga se define la operación de selección sobre una medida de necesidad, salvo que se utiliza un comparador Θ_{N_τ} asociado a la misma.

Las expresiones asociadas a las operaciones de *join* y *proyección* se llevan a cabo teniendo en cuenta las mismas consideraciones, y son las siguientes:

Supongamos una relación n-aria R y que queremos proyectarla sobre sus k primeros atributos.

$$R[A_1, A_2, \ldots, A_k] \equiv < x_1, x_2, \ldots, x_k / \tau_1(x_1), \tau_2(x_2), \ldots, \tau_k(x_k) \ \mid \ \exists (w_{k+1}, \ldots, w_m),$$

$$(R(x_1, x_2, \ldots, x_k, w_{k+1}, \ldots, w_m) \wedge$$

$$(MD(x_1) \wedge \forall (x, v_{k+1}, \ldots, v_m), \ (R(x, x_2, \ldots, v_{k+1}, \ldots, v_m) \longrightarrow \geq (x_1, x))) \vee$$

$$\ldots\ldots$$

$$\ldots\ldots$$

$$\ldots\ldots$$

$$(MD(x_k) \wedge \forall (x, v_{k+1}, \ldots, v_m), \ (R(x_1, x_2, \ldots, x_{k-1}, x, v_{k+1}, \ldots, v_m) \longrightarrow \geq (x_k, x)))) >$$

En esta sección, hemos identificado los modelos de BDRD de la literatura como casos particulares de nuestro modelo tanto desde el punto de vista de la estructura de los datos como del lenguaje de consulta, lo cual confirma la generalidad y versatilidad de nuestra aproximación lógica.

La descripción formal detallada de cómo quedan ambos modelos englobados en nuestro modelo lógico general, junto con todos los operadores mencionados, puede encontrarse en [122].

3.4 BDD como Teoría Relacional Difusa

La aproximación de una BDD desde el punto de vista de la Teoría de Pruebas es bastante directa, teniendo en cuenta la definición de IR dada. Como ya indicamos en la sección anterior, una BDD es un triple (L, I, RI) en el que L es un lenguaje

relacional difuso, I es una interpretación relacional difusa y RI es un conjunto de reglas de integridad, al que añadiremos, en este caso, todas las reglas necesarias para definir y manipular los tipos difusos, esto es, las reglas R1 a R8 especificadas en def. 3.4. Llamaremos CRI (conjunto de reglas de integridad) a $\{R1, \ldots, R8\} \cup RI$.

Dado que un LRD es un lenguaje relacional en el sentido de Reiter (pero con algunas propiedades adicionales), y dado que una IRD es una interpretación relacional en el sentido de Reiter (con algunas reglas de integridad adicionales), todos los resultados obtenidos en [102] serán aplicables a nuestras definiciones. En particular, se verificará la siguiente propiedad:

Propiedad:

Sea $DB = (L, I, RI)$ una BDD. Entonces, T es una **Teoría Relacional Difusa** si I es el único modelo de T. De acuerdo con este resultado, la teoría T contendrá los siguientes axiomas:

1. *Axioma de dominio cerrado.*
2. *Axioma de nombre único.*
3. *Axiomas de igualdad*, tal y como se especifican en la sección 1.2.2.
4. Para cada predicado P, la *extensión del predicado* y el *axioma de completitud* si el conjunto C_p (definido en 3.9) no es el conjunto vacío, o la negación de dicho predicado, en otro caso.
5. Las únicas fbf de T son las mencionadas en los apartados anteriores.

Nótese, que al ser I el único modelo de T, los conceptos de *verdad* en I y de *demostrabilidad* en T son equivalentes. Esta equivalencia, nos permite definir el lenguaje de consulta desde la perspectiva de la teoría de pruebas, como sigue.

Si $Q =< X/\Phi \mid W(X) >$ es una consulta para el lenguaje L, entonces una tupla $C = (c_1, c_2, \ldots, c_n)$ es una *respuesta para* Q respecto a la BD si, y sólo si:

- $T \vdash \tau_i(c_i),\ \forall i \in \{1, 2, \ldots, n\}$

– $T \vdash W(C)$

Todo ello, nos lleva a establecer las siguientes definiciones:

Definición 3.6 .

Sea $L = (A, W)$ un LRD, el conjunto $T \subseteq W$ es una **Teoría Relacional Difusa** *si, y sólo si:*

(i) Es una teoría relacional según la definición de Reiter.

(ii) Existe una interpretación relacional difusa, que es el único modelo para ella.

(iii) Incluye todas las reglas de integridad de los tipos difusos de L (R1 a R8).

Definición 3.7 .

Una **BDD según el Enfoque de las Demostraciones**, *es un triple (L, T, RI), en el que T es una teoría relacional difusa y RI es un conjunto específico de reglas de integridad. Llamaremos a esta estructura* Base de Datos Lógica Difusa.

3.4.1 Ejemplo

Consideremos la tabla 3.1 como una interpretación relacional difusa.

La teoría relacional difusa asociada consta de las siguientes fbf:

1. *Axioma de Dominio Cerrado*:

$$\forall x, = (x, Jose) \vee = (x, Maria) \vee \ldots \vee = (x, Susana) \vee$$

$$= (x, Muy\ Joven) \vee = (x, 14) \vee \ldots \vee = (x, Media) \vee$$

$$\ldots\ldots$$

$$= (x, Regular = \vee = (x, Bueno) \vee \ldots \vee = (x, Ejemplar)$$

2. *Axioma de Nombre Unico*:

$$\neg = (Jose, Maria);\ \neg = (Jose, Antonio); \ldots \neg = (Jose, Ejemplar)$$

$$\neg = (Maria, Antonio);\ \neg = (Maria, Carlos); \ldots \neg = (Maria, Ejemplar)$$

$$\ldots\ldots$$

$$\neg = (Regular, Bueno);\ \neg = (Regular, Muy\ Bueno); \ldots \neg = (Regular, Ejemplar)$$

3. *Axiomas de Igualdad*:

   Se especificarán las propiedades de reflexividad, conmutatividad y transitividad, tal y como se hace en 1.2.2. El *principio de sustitución* queda como sigue:

$$\forall\ (x, y)\ Nombre(x) \wedge = (x, y) \longrightarrow Nombre(y)$$

$$\forall\ (x, y)\ Edad(x) \wedge = (x, y) \longrightarrow Edad(y)$$

$$\ldots\ldots$$

$$\forall\ (x, y, z, t, x_1, y_1, z_1, t_1)\ Alumno(x, y, z, t) \wedge = (x, x_1) \wedge = (y, y_1) \wedge = (z, z_1) \wedge$$
$$= (t, t_1) \longrightarrow Alumno(x_1, y_1, z_1, t_1)$$

4. *Extensión de los Predicados y Axiomas de Completitud*

   Axiomas para los predicados de la BD

$$Nombre(Jose),\ Nombre(Maria), \ldots, Nombre(Susana)$$

$$Edad(Muy\ Joven),\ Edad(14), \ldots, Edad(Media)$$

$$Curso(1),\ Curso(2), \ldots, Curso(5)$$

$$Comp(Regular),\ Comp(Bueno), \ldots, Comp(Ejemplar)$$

$$\forall x,\ Nombre(x) \longrightarrow = (x, Jose) \vee \ldots \vee = (x, Susana)$$

$$\forall x,\ Edad(x) \longrightarrow = (x, MuyJoven) \vee \ldots \vee = (x, Media)$$

$$\forall x,\ Curso(x) \longrightarrow = (x, 1) \vee \ldots \vee = (x, 5)$$

$$\forall x,\ Comp(x) \longrightarrow = (x, Regular) \vee \ldots \vee = (x, Ejemplar)$$

$$\forall\ (x, y, z, t)\ Alumno(x, y, z, t) \longrightarrow$$

$$(= (x, Jose) \wedge = (y, MuyJoven) \wedge = (z, 1) \wedge = (t, Regular)) \vee$$

$$\ldots\ldots$$

$$\vee\ (= (x, Susana) \wedge = (y, Media) \wedge = (z, 5) \wedge = (t, Ejemplar))$$

Axiomas para el operador $\cong_\tau$

$$\cong_{Comp} (Malo, Regular, 0.9);\ \cong_{Comp} (Malo, Bueno, 0.4);$$

$$\cong_{Comp} (Regular, Bueno, 0.7);\ \cong_{Comp} (Regular, Muy\ Bueno, 0.3);$$

$$\cong_{Comp} (Bueno, Muy\ Bueno, 0.8);\ \cong_{Comp} (Bueno, Ejemplar, 0.5);$$

$$\cong_{Comp} (Muy\ Bueno, Ejemplar, 0.9)$$

$$\forall\ (x, y, z),\ \cong_{Comp} (x, y, z) \longrightarrow (= (x, Malo) \wedge = (y, Regular) \wedge = (z, 0.9)) \vee$$

$$\ldots\ldots$$

$$\vee\ (= (x, Bueno) \wedge = (y, Ejemplar) \wedge = (z, 0.5))$$

De acuerdo con la definición 3.4, para que esta teoría relacional sea difusa, será necesario incluir las reglas asociadas a los tipos difusos y la semantica del tipo MD.

Como puede verse, el modelo lógico para BDRD que hemos desarrollado y que aparece en [123], tiene la misma estructura que el correspondiente desarrollado por Reiter para BDR clásicas, salvo que se hace necesario utilizar un lenguaje algo más complejo y una serie de reglas de integridad asociadas a los elementos difusos. Nos queda, sin embargo, por tratar uno de los principales problemas del modelo relacional en general, el de representar la información imprecisa de tipo disyuntivo y los valores nulos.

3.5 Tratamiento de la Información Incompleta

En esta sección trataremos de ver cómo pueden realizarse una representación adecuada y una manipulación coherente de la información de tipo disyuntivo y de los distintos tipos de valores nulos que pueden aparecer.

Estos dos problemas, no tienen una buena solución en la aproximación de una BD como interpretación relacional, ya que, en la mayoría de los casos, su representación nos obliga a considerar varias interpretaciones a la vez. Por ejemplo, en la representación de un hecho como: *"Pedro tiene 14 ó 15 años"*, habría que considerar dos interpretaciones, esto es, una en la que se incluya *" Pedro tiene 14 años"* y otra en la que se incluya *"Pedro tiene 15 años"*.

Para el caso de los valores nulos, el problema sería el mismo, ya que el valor nulo como *desconocido* queda reducido a una disyunción entre todos los elementos posibles del dominio. Si además no hay certeza de que sea un valor del dominio, el problema se agudiza, ya que hay que incluir información ficticia al sistema y considerar un tercer grado de verdad, el *desconocido*, para la interpetación.

Una justificación más formal de estos problemas puede encontrarse en [102].

Todo ello, nos lleva a tratar de dar solución a este problema desde el marco del enfoque de las demostraciones. Por tanto, partiendo de esta aproximación y teniendo en cuenta la nueva definición lógica dada para BDD, trataremos de dar solución a una serie de problemas que ya apuntaba Reiter, y a otros adicionales derivados del carácter difuso de algunos atributos.

3.5.1 Tratamiento de la Información Disyuntiva

Se trata de representar información de la forma:

$$Q(x) \vee P(y)$$

esto es, no se conoce si se da $Q(x)$ o se da $P(x)$; sólo hay seguridad de que alguno de ellos (o ambos) es cierto. En este punto, habría que distinguir entre dos posibles

interpretaciones de la disyunción:

- **Información Disyuntiva No Exclusiva***: En este caso la fórmula $Q(x) \vee P(y)$ es cierta si $Q(x)$ es verdad o bien $P(y)$ es verdad, o son ambos predicados verdad. Por ejemplo, la fórmula $matriculado(Pedro, Calculo) \vee matriculado(Pedro, Computabilidad)$ será cierta si Pedro está matriculado en cálculo, si está matriculado en computabilidad o si está matriculado en ambas. Sin embargo, hemos supuesto que no hay ningún *"impedimento"* para que se den ambas cosas a la vez.

- **Información Disyuntiva Exclusiva**: Hay veces en que ambos predicados no pueden hacerse verdad simultáneamente desde el punto de vista de su significado real o por restricciones del sistema; por ejemplo $Edad(Pedro, 15) \vee Edad(Pedro, 16)$ sería un caso de disyunción exclusiva ya que ambas cosas no son satisfacibles simultáneamente desde el punto de vista de la realidad que se intenta modelizar con dicha fórmula, y no debería permitirse tampoco desde el punto de vista sintáctico que ambas cosas pudieran probarse simultáneamente.

◇ Análisis del Caso Clásico

Antes de formalizar el tratamiento de la información incompleta desde el marco de las BD lógicas difusas, vamos a destacar y comentar los detalles más relevantes del caso clásico en lo que se refiere al tratamiento de este mismo problema.

Para Reiter [102], la solución a este problema tiene el siguiente planteamiento:

Supongamos que hay que representar el hecho anterior. Lo expresaremos como:

$$Edad(Pedro, 14) \vee Edad(Pedro, 15)$$

suponiendo que tanto el predicado $Edad$ como los tipos de los argumentos y su número son correctos.

En esta situación, bastaría con ampliar el *axioma de completitud* para el predicado

*Los predicados involucrados no son mutuamente excluyentes

Edad (en general para cualquier predicado involucrado), de manera que incluya las dos nuevas posibilidades, esto es:

$$\forall\ (x,y)\ Edad(x,y) \longrightarrow \ldots \vee (= (x, Pedro) \wedge = (y, 14)) \vee (= (x, Pedro) \wedge = (y, 15))$$

Estas consideraciones llevaron a Reiter a una generalización del concepto de BD relacional con información disyuntiva, como sigue:

Definición 3.8 .

Sea $R = (A, W)$ una teoría relacional. Una fbf de R se dice que es una **cláusula positiva robusta** *si, y sólo si, tiene la forma $A_1 \vee A_2 \vee \ldots \vee A_n$ donde las A_i son fórmulas atómicas robustas cuyo predicado no es el de igualdad.*

Definición 3.9 .

Una teoría de primer orden $T' \subseteq W$ es una **teoría relacional generalizada** *de T si, y sólo, si satisface las siguientes propiedades:*

1. *Incluye el axioma de dominio cerrado.*
2. *Incluye el axioma de nombre único.*
3. *Incluye las extensiones de todos los predicados.*
4. *Contiene los axiomas de igualdad (transitividad, reflexividad, conmutatividad y sustitución de términos iguales).*
5. *Sea $\Delta \subseteq W$ un conjunto de cláusulas positivas robustas y sea*

$$C_p = \{(c_1, \ldots, c_m) \mid \exists A_1 \vee \ldots \vee A_r \in \Delta\ y$$

$$\exists\ i \in \{1, \ldots, r\}\ verificando\ P(c_1, \ldots, c_m) = A_i\}$$

En esta situación, si $C_p = \{(c_1^1, \ldots, c_m^1), \ldots, (c_1^r, \ldots, c_m^r)\}$, entonces el axioma de completitud incluye las fbf de Δ y el siguiente axioma de completitud para P:

$$\forall\ (x_1, \ldots, x_m),\ P(x_1, \ldots, x_m) \longrightarrow$$

$$(= (x_1, c_1^1) \wedge, \ldots, \wedge = (x_m, c_m^1)) \vee \ldots \vee (= (x_1, c_1^r) \wedge, \ldots, \wedge = (x_m, c_m^r))$$

Si, por el contrario, $C_p = \emptyset$*, entonces el axioma de completitud de* P *tiene la forma:*

$$\forall\ (x_1, \ldots, x_m),\ \neg P(x_1, \ldots, x_m)$$

6. Las únicas fbf de T' son las que se especifican en los apartados anteriores.

Nótese que esta definición tiene como caso particular la definición de teoría relacional dada en 3.4.

Teorema 3.1 .

Toda Teoría Relacional Generalizada T *es* consistente.

Demostración*:*

Aunque puede encontrarse en [102], pasamos a comentarla por su interés en lo sucesivo. Este teorema se prueba construyendo un modelo para la teoría relacional generalizada $T \subseteq W$*. Definimos, para el lenguaje relacional* $R = (A, W)$*, una* interpretación $I = (D, K, E)$ *cuyo dominio* $D = \{c_1, c_2, \ldots, c_n\}$ *es igual al de las constantes de* A *y sea* $E(=) = \{(c, c) \mid c \in D\}$*. En esta situación,* I *satisface los axiomas de dominio cerrado, nombre único y de igualdad de* T*. Por último, si para cada símbolo de predicado de* A *distinto del de igualdad definimos* $E(P) = \{\vec{c} \mid \vec{c} \in C_p\}$*, entonces* I *satisface todas las fbf de* Δ*, así como los axiomas de completitud de todos los predicados de* A *distintos del de igualdad. Así pues, podemos concluir que* I *es un modelo de* T*.*

Esta teoría, si bien permite deducir la disyunción/es considerada/s inicialmente, no tiene un buen comportamiento para el caso de las BDD. Esta afirmación la justificamos a continuación. Volviendo al marco de las BDD, podría ser deseable que, ante una consulta del tipo *"Dime los nombres de aquellos alumnos que tengan aproximadamente 15 años"*, el sistema diese entre sus resultados a *Pedro*, mientras que para la consulta precisa *"Dime los nombres de los alumnos que tienen 15 años"*, *Pedro* no fuese

considerado, puesto que no hay seguridad alguna de que lo verifique. Teniendo esto en cuenta, la aproximación de Reiter daría resultados deseados para el caso de consultas completamente precisas, pero no es una aproximación completa en el sentido de las BDD.

A nuestro entender, la solución a este problema pasa por su solución previa desde el marco de las BDD, para luego ser contemplada desde el punto de vista de las BD Lógicas Difusas.

◇ Interpretación de la Información Disyuntiva en el Marco de las BDD

Este tipo de información es aquella en la que un dato toma uno o varios valores de entre los de un conjunto, pero no conocemos cuál o cuáles.

Empezaremos por establecer una semántica concreta para afirmaciones del tipo *"Pedro es alto"*, donde *alto* viene dado, por ejemplo, por la siguiente distribución de posibilidad:

$$alto = 1.80/0.4 + 1.81/0.5 + 1.82/0.6 + 1.83/1 + 1.84/1 + \ldots + 1.95/0.4$$

Como sabemos, cada persona tiene asociado un **único** valor real que represente su altura, dentro de un rango de posibles valores para dicho atributo. Dado que el difuso *"alto"* es una restricción difusa sobre los valores a tomar por la variable *altura de Pedro* [136], en este sentido, la afirmación de que *"Pedro es alto"*, puede traducirse por la siguiente:

"La altura de Pedro es 1.80 con posibilidad 0.4 $\vee$ La altura de Pedro es 1.81 con posibilidad 0.5 $\vee$ ..... $\vee$ La altura de Pedro es 1.95 con posibilidad 0.4"

pero su altura sólo toma uno de los posibles valores que componen el conjunto difuso *alto*, con lo que, no sólo se trata de información disyuntiva, sino que, además, en este caso, es de tipo exclusivo (sólo se da una de las opciones).

En general, si el valor de un atributo en una tupla t es una distribución de posibilidad del tipo:

$$v_1/p_1 + v_2/p_2 + \ldots + v_n/p_n$$

entenderemos que dicho atributo puede tomar cualquiera de los valores v_i, con posibilidad p_i.

De hecho, no tendría sentido que su significado fuera otro, ya que la conjunción se expresa (en la mayoría de los modelos de BD) mediante la duplicación de tuplas*.

Si nuestra información es que *"Pedro está matriculado en Análisis o en Programación"*, su representación (en una supuesta tabla de alumnos y edades) sería:

Pedro	$Analisis/1 + Programacion/1$

Si además está matriculado en otras asignaturas, se añadiría una tupla por cada una de ellas:

Pedro	Bases de Datos
Pedro	Computabilidad
...	...

Una vez establecidos estos conceptos, tenemos que la información disyuntiva, en el marco de las BDD, vendrá dada por una distribución de posibilidad, con o sin nombre asociado. Dar una representación a este tipo de información en el marco de las BD lógicas difusas, considerando la BD como una teoría relacional difusa, es casi directo; no tenemos más que utilizar correctamente los predicados $NOP_\tau(-)$ $POS_\tau(-,-,-)$, siguiendo las restricciones de integridad que para ellos se han introducido. Como puede verse, estas restricciones sólo nos obligan a que dichas distribuciones lleven un nombre asociado, que habrá de ser incluido en el axioma de dominio cerrado. Para el manejo de dicha información, se incluirán, además, las instancias correspondientes del predicado $\cong$ (*aproximadamente igual*).

*Si tuviéramos que representar el hecho de que Pedro habla francés e inglés, aparecerían dos tuplas, una en la que el idioma hablado es francés y otra, en la que es inglés

Puede verse fácilmente, que este nuevo planteamiento para la información disyuntiva no plantea los problemas anteriormente mencionados para las consultas no precisas. Veamos un ejemplo.

Supongamos que en la tabla 3.1 aparece la tupla siguiente:

Pedro	E=14/1+15/1	4	Bueno

Esta tupla, daría lugar a los siguientes predicados:

$$NOP_{Edad}(E)$$

$$POS_{Edad}(E, 14, 1), POS_{Edad}(E, 15, 1)$$

$$Alumno(Pedro, E, 4, Bueno)$$

$$\cong_{Edad}(E, 14, 1), \cong_{Edad}(E, 15, 1), \cong_{Edad}(E, Joven, 0.9), \cong_{Edad}(E, Muy\ Joven, 0.8), \ldots$$

En esta situación, si queremos realizar la consulta *"Encontrar aquellos alumnos que tengan más o menos 14 años con posibilidad 0.9"*, podremos hacerlo de la siguiente forma:

$$< x/Nombre, y/Edad, z/Curso, t/Comp, w/MD \ \mid$$

$$Alumno(x, y, z, t) \wedge \cong_{Edad}(y, 14, w) \wedge \geq (w, 0.9) >$$

donde entraría a formar parte de la solución la tupla anteriormente introducida. Nótese que, si en vez de usar el operador $\cong_{\tau}$, hubiésemos usado el $=$ (consulta precisa), esta última tupla no formaría parte de la solución. De hecho, según la tabla 3.1 la única solución hubiese sido la tupla:

Maria	14	1	Bueno

Nótese que el hecho de que la información disyuntiva sea o no exclusiva, es una cuestión que depende del significado de cada atributo. Por ejemplo, la afirmación

$$Edad(Pedro, 14) \vee Edad(Pedro, 15)$$

no admite, desde el punto de vista semántico, la posibilidad de que ambas afirmaciones sean ciertas simultáneamente, mientras que la afirmación

$$Matriculado(Pedro, Algebra) \vee Matriculado(Pedro, Computabilidad)$$

si lo admitiría. Dado que la representación para la información disyuntiva es única e incluye esta última posibilidad, en el caso de que la información sea disyuntiva exclusiva, será necesario introducir las reglas de integridad adecuadas. Por ejemplo, en el caso de la edad podría ser la siguiente:

$$\forall x,\ Edad(x, y) \longrightarrow \neg\exists z,\ (Edad(x, z) \wedge Edad(x, y) \wedge \neg = (y, z))$$

En muchos casos, el carácter exclusivo de la disyunción viene ya determinado por las reglas que definen la dependencia funcional de la llave primaria de una relación. Por ejemplo, si aparece la siguiente regla de integridad:

$$\forall\ (x, y_1, y_2, z_1, z_2, w_1, w_2),\ Alumno(x, y_1, z_1, w_1) \wedge Alumno(x, y_2, z_2, w_2) \longrightarrow = (y_1, y_2)$$

ya no sería necesaria otra regla adicional para especificar el carácter exclusivo del atributo y.

Sin embargo, si sólo tenemos en consideración lo aquí expuesto, no puede demostrarse a partir de la teoría así construida, la afirmación inicial completa, por ejemplo, que *"Pedro tiene 14 ó 15 años"*. Para que ello sea posible, será necesario considerar, adicionalmente, la modificación del axioma de completitud tal y como lo hace Reiter, pero en el marco de las BDD, y definir el concepto de *Teoría Relacional Difusa Generalizada* como sigue.

Definición 3.10 .

Una teoría relacional difusa *de primer orden $T' \subseteq W$ es una* **Teoría Relacional Difusa Generalizada** *de T si, y sólo si, satisface las mismas propiedades enunciadas en la definición 3.9 para el caso clásico.*

3.5.2 Tratamiento del Valor Nulo

El tratamiento y representación de valores nulos en una BD, es un problema muy estudiado en todos los modelos de BD. Aunque los sistemas de BD tradicionales sólo suelen considerar un tipo de valor nulo, que además es común para cualquier atributo, existen distintos tipos de valores nulos según el significado que se les atribuya. En general, podemos distinguir tres tipos distintos de valores nulos:

- *Desconocido*: No se conoce el valor que toma una tupla para un determinado atributo, pero seguro que es uno de los valores del dominio subyacente. Más formalmente, si $D_\tau = \{d_{\tau_1}, d_{\tau_2}, \ldots, d_{\tau_n}\}$ es el conjunto de valores permitidos para un atributo τ, el valor *desconocido* como valor del atributo τ significa que puede ser cualquiera de los del dominio, pero sólo uno de ellos, esto es, que el valor real para dicho dato puede ser d_{τ_1} ó d_{τ_2} ó .... ó d_{τ_n}. Nos encontramos, claramente, ante un caso de información disyuntiva exclusiva.

- *Inaplicable*: En este caso, no existe ningún valor del dominio posible para dicho dato, esto es, no tiene sentido alguno asociar un valor a ese atributo en la tupla. Situaciones como ésta se dan cuando, por ejemplo, se pretende almacenar el color de pelo de una persona calva o el nombre del cónyuge para una persona soltera.

- *Nulo*: Es el caso más extremo de desconocimiento, ya que supone que no se conoce cuál de las dos situaciones anteriores es la que se da, si el desconocimiento de un valor o su inaplicabilidad.

Una clasificación análoga y la manipulación correspondiente de los distintos tipos de información incompleta, aparece en el modelo relacional RM/V2 debido a Codd (véase [31]) y en el modelo FPDB que aparece en [76].

En el primero de ellos (Codd), la representación dada para este tipo de información está basada en la adición de un campo axiliar a cada atributo, indicando si se trata de un valor inaplicable, de un valor desconocido o de un valor aplicable y conocido. En ningún caso se permite representar la semántica de los dos primeros (se dar por supuesta) ni se habla del desconocimiento total, es decir, de lo que hemos llamado en nuestra clasificación *nulo*.

En el segundo modelo (debido a Li y Liu), se consideran los tres tipos anteriormente mencionados pero desde una perspectiva distinta (respecto a los nulos) de la que vamos a considerar.

En esta sección, trataremos de incorporar los tres tipos de nulos mencionados en el marco de las BD Lógicas Difusas. Hay que hacer notar, que no existe ningún modelo en el que se manejen todos ellos de forma homogénea dentro del propio modelo (exceptuando el modelo FPDB para los Inaplicables y Desconocidos), sino que se hace por medio de la incorporación de estructuras y elementos *"ad hoc"* para su manejo. Un estudio sobre la semántica del valor nulo como respuesta a una consulta puede verse en [85].

◇ El Nulo como Desconocido

Como hemos indicado anteriormente, se trata de información disyuntiva exclusiva, en la que se abarca todo el abanico de valores permitidos para el dominio del tipo (atributo) considerado. Igual que en el caso de la disyunción, dentro del ámbito de las BDD, sería deseable que, para ciertas consultas no precisas, fuera considerado el dato *desconocido* como posible respuesta. En este sentido, el tratamiento que propone Reiter para el manejo de este tipo de dato, sigue siendo restrictivo.

Siguiendo una notación análoga a la utilizada para el caso de la disyunción, un dato de este tipo quedará representado de la siguiente forma:

Sea $D_\tau = \{d_{\tau_1}, d_{\tau_2}, \ldots, d_{\tau_n}\}$ el dominio de τ. El valor nulo como desconocido vendrá dado, pues, por la siguiente distribución

$$d_{\tau_1}/1 + d_{\tau_2}/1 + \ldots + d_{\tau_n}/1$$

esto es, la que asigna igual posibilidad a todos los valores del dominio.

Dada la representación que hemos considerado, un dato de este tipo quedará expresado en la teoría como cualquier otra distribución, esto es, por medio del predicado $POS\tau$. Es importante señalar que para un mismo tipo (o lo que es igual, para un mismo atributo) los posibles valores nulos que aparezcan tienen la misma forma; por

ello, será suficiente con definir un valor de este tipo por cada tipo o atributo considerado. Notaremos a este valor por Unk_τ.

Esta idea es, hasta cierto punto, análoga a lo que Date define como *valores por defecto* [35] donde, de cada dominio, se toma un valor especial llamado *valor por defecto*, que se asigna automáticamente cuando no se suministra el valor de un atributo en una tupla. Sin embargo, no se suministra semántica alguna para estos valores, no quedando clara cuál es la causa de la ausencia del valor omitido, esto es, este valor por defecto se aplica tanto en el sentido de valores desconocidos como el de valores no aplicables.

En el caso de FPDB se utiliza un único valor desconocido para todos los atributos. Esta aproximación no es semánticamente correcta dentro de nuestro modelo lógico, puesto que las distribuciones de posibilidad admisibles para un atributo deben estar necesariamente definidas sobre el dominio de dicho atributo, en particular la distribución que define el valor desconocido* (lo mismo sucede con el valor *Inaplicable*).

Así pues, la representación adoptada para datos desconocidos, nos lleva a modificar la teoría subyacente, en el siguiente sentido:

- Habrá que *extender el axioma de dominio cerrado* e incluir, para cada unos de los tipos existentes, el valor *desconocido* correspondiente. Sean $c_1, c_2, \ldots, c_m$ todas las constantes del dominio de la BD de partida, y sean $\tau_1, \tau_2, \ldots, \tau_n$ los tipos existentes. Entonces, el axioma de dominio cerrado tiene la forma:

$$\forall x, \ = (x, c_1) \vee = (x, c_2) \vee \ldots \vee$$
$$= (x, c_m) \vee = (x, Unk_{\tau_1}) \vee = (x, Unk_{\tau_2})$$
$$\vee \ldots \vee = (x, Unk_{\tau_n})$$

- Teniendo en cuenta la extensión del axioma de dominio cerrado, también será necesario extender el correspondiente axioma de completitud para el predicado involucrado.

  Supongamos que se desea representar el siguiente predicado $P(c_1, c_2, \ldots, Unk_{\tau_i}, \ldots, c_m)$ correspondiente a una tupla de una relación. Entoces, la extensión de

*Es probable que en una implementación concreta del modelo, se reduzcan todos estos valores desconocidos a un único valor desconocido, por simplicidad

dicho predicado en la teoría podría realizarse tal y como se indica, gracias a la representación que para los valores desconocidos hemos dado, esto es:

$$\forall(x_1, x_2, \ldots, x_m),\ P(x_1, x_2, \ldots, x_m) \longrightarrow .....\vee$$

$$(= (x_1, c_1) \wedge \ldots \wedge = (x_i, Unk_{\tau_i}) \wedge \ldots \wedge = (x_m, c_m))$$

Como consecuencia inmediata de la definición dada para el valor *desconocido*, también se verán modificadas las extensiones de los predicados NOP_τ, que deberá incluir:

$$NOP_{\tau_1}(Unk_{\tau_1}), NOP_{\tau_2}(Unk_{\tau_2}), \ldots, NOP_{\tau_n}(Unk_{\tau_n})$$

y $\cong_\tau$, que deberá incluir:

$$\cong_\tau (Unk_\tau, x, 1)\ \forall x\ verificando\ \tau(x)$$

– Nótese, que no debe modificarse el axioma de nombre único para incluir *todas* las nuevas constantes del dominio, ya que éstas toman, precisamente, valores ya existentes del dominio anterior, aunque no sepamos cuáles. En todo caso, deberán considerarse sólo aquellas desigualdades de las que se tiene seguridad, por ejemplo:

$$\neg = (Unk_{\tau_i}, c_j)\ para\ algun\ 1 \leq i \leq n,\ 1 \leq j \leq m$$

y, por supuesto, las desigualdades:

$$\forall i, \forall j, \neg = (Unk_{\tau_i}, Unk_{\tau_j})$$

Esta representación permite, adicionalmente, expresar cierto tipo de información parcialmente desconocida, esto es, de la que se conocen ciertos detalles, como por ejemplo *"Juan está matriculado en una asignatura, pero no es ni Algebra ni Cálculo"*, que quedaría expresada como:

$$Matriculado(Juan, Unk_{Asg}) \wedge \neg Matriculado(Juan, Analisis)$$

$$\wedge \neg Matriculado(Juan, Calculo)$$

Todas estas consideraciones difieren de las enunciadas por Reiter para el mismo caso, ya que él considera un nombre diferente para cada uno de los valores nulos, independientemente de que éstos pertenezcan o no al mismo dominio y se apliquen al mismo atributo. De manera análoga, en [65], se introducen símbolos especiales llamados *"variables"*, para representar valores nulos. Cada variable se refiere exactamente al mismo valor, por lo que habrá tantas variables como nulos *diferentes*. También en [43] se parte de esta consideración.

En esta aproximación, así como en la de Reiter, la única diferencia formal entre un valor *desconocido* y cualquier otro valor del dominio es que, para el primero, estarán ausentes de la teoría algunos de los axiomas de nombre único. Todas estas ideas pueden ser formalizadas como sigue.

Definición 3.11 .

Sea $T = (A, W)$ una teoría relacional difusa, en la que A está particionado en dos subconjuntos disjuntos de constantes $C = \{c_1, c_2, \ldots, c_m\}$ y $N = \{Unk_{\tau_1}, Unk_{\tau_2}, \ldots, Unk_{\tau_m}\}$, donde N puede ser el conjunto vacío. Cada uno de los elementos de N recibe el nombre de **valor nulo desconocido**. *La Teoría Relacional Difusa $T \subseteq W$ es una* **Teoría Relacional Difusa Generalizada con Valores Desconocidos** *si, y sólo si, satisface las siguientes propiedades:*

1. *T contiene el axioma de dominio cerrado extendido*

$$\forall x, = (x, c_1) \vee = (x, c_2) \vee \ldots \vee = (x, c_m) \vee = (x, Unk_{\tau_1}) \vee = (x, Unk_{\tau_2})$$

$$\vee \ldots \vee = (x, Unk_{\tau_n})$$

2. *El axioma de nombre único extendido, incluyendo*

$$\neg = (Unk_{\tau_i}, c_j) \text{ para algun } 1 \leq i \leq n,\ 1 \leq j \leq m$$

$$\forall i, \forall j, \neg = (Unk_{\tau_i}, Unk_{\tau_j})$$

3. T contiene los axiomas de la igualdad.

4. T contiene las extensiones de todos los predicados.

5. Sea $\Delta \subseteq W$ *un conjunto de cláusulas positivas robustas y sea*

$$C_p = \{(c_1, \ldots, c_m) \mid \exists\ A_1 \vee \ldots \vee A_r \in \Delta,\ y\ \exists i,\ tales\ que\ A_i = P(c_1, \ldots, c_m)\}$$

En esta situación, si

$$C_p = \{(c_1^1, \ldots, Unk_{\tau_i}, \ldots, c_m^1), \ldots, (c_1^r, \ldots, c_m^r)\}$$

entonces el axioma de completitud incluye las fbf de Δ *y el siguiente axioma de completitud para P:*

$$\forall (x_1, x_2, \ldots, x_m),\ P(x_1, x_2, \ldots, x_m) \longrightarrow .....\vee$$

$$(= (x_1, c_1) \wedge \ldots \wedge = (x_i, Unk_{\tau_i}) \wedge \ldots \wedge = (x_m, c_m))$$

Si, por el contrario, $C_p = \emptyset$*, entonces el axioma de completitud de P tiene la forma:*

$$\forall\ (x_1, \ldots, x_m),\ \neg P(x_1, \ldots, x_m)$$

Como puede verse, este axioma no difiere del considerado para el caso de la disyunción, salvo que pueden incluirse nombres de valores nulos entre los argumentos de los predicados.

6. Las únicas fbf de T son las que se construyen según los puntos anteriores.

En el caso de definirse operadores como $>$, $>$, $\leq$,... en un determinado dominio, será necesario incluir las restricciones correspondientes para los valores Unk_τ. Por ejemplo, si se define el operador $<_\tau$ para un dominio τ, podría incluirse, entre otras, la regla:

$$<_\tau (x, y, z) \wedge (= (x, Unk_\tau) \vee = (y, Unk_\tau)) \longrightarrow = (z, Unk_{MD})$$

Definición 3.12 .

Una **BD Relacional Difusa Generalizada con Valores Desconocidos** *es un triple* (R, T, RI) *donde* R *y* T *son como en la definición anterior, y* $RI \subseteq W$ *es un conjunto de restricciones de integridad.*

Teorema 3.2 *Toda teoría relacional difusa generalizada con valores desconocidos* T *es* consistente.

Demostración*:*

Como hemos visto, el único axioma que no verifica T *para ser una teoría relacional generalizada es el de nombre único. Con la idea que este axioma también sea verificado, extenderemos* T *de manera que incluya todas las desigualdades de la forma:*

$$\neg = (Unk_\tau, c_j)\ \forall j$$

$$\neg = (Unk_{\tau_i}, Unk_{\tau_j})\ \forall\ i, j\ \mid\ i \neq j)$$

además de las referentes a las constantes c_i *del dominio. De esta manera, tenemos una teoría relacional generalizada. Como por el teorema 3.1 sabemos una teoría así construida es consistente, también lo será, en particular, cualquier subconjunto de ella, concretamente aquél al que le falten algunos de los axiomas de igualdad, que es el caso de* T*.*

Como ya indicamos al comienzo de esta sección, esta representación para el valor nulo como *desconocido*, nos permite considerar (o no) tuplas en las que dicho valor aparece, cuando se realiza una consulta no precisa a la BD. Veamos un ejemplo.

Consideremos la tabla de datos 3.2 y supongamos que se realizan las siguientes consultas:

* *"Encontrar aquellos alumnos que tengan alrededor de 14 años con posibilidad 0.8"*

$$< x/Nombre, y/Edad, z/Curso, t/Comp, w/MD\ \mid$$

Nombre	**Edad**	**Curso**	**Comportamiento**
Jose	Unk_{Edad}	1	Regular
Maria	14	1	Bueno
Antonio	Joven	2	Unk_{Comp}
Carlos	16	3	Malo
Susana	Media	5	Ejemplar

Tabla 3.2. Tabla ALUMNOS (b)

Jose	Unk_{Edad} $(w = 1)$	1	Regular
Maria	14 $(w = 1)$	1	Bueno
Antonio	Joven $(w = 0.6)$	2	Unk_{Comp}

Tabla 3.3. Alumnos de aprox. 14 años con posibilidad 0.8

$$Alumno(x, y, z, t) \wedge \cong_{Edad} (y, 14, w) \wedge \geq (w, 0.8) >$$

que dará como resultado la tabla 3.3.

* *"Encontrar aquellos alumnos de los que se conozca la edad y que tengan alrededor de 14 años con posibilidad 0.8"*

$$< x/Nombre, y/Edad, z/Curso, t/Comp, w/MD \ \mid \ Alumno(x, y, z, t) \wedge$$

$$\neg = (y, Unk_{Edad}) \wedge \cong_{Edad} (y, 14, w) \wedge \geq (w, 0.8) >$$

que dará como resultado la tabla 3.4.

Maria	14 $(w = 1)$	1	Bueno
Antonio	Joven $(w = 0.6)$	2	Unk_{Comp}

Tabla 3.4. Alumnos de edad conocida aprox. 14 años con posibilidad 0.8

◇ El Nulo como No Aplicable

Se trata de dar cabida en nuestro modelo a datos que no son susceptibles de tomar ningún valor del dominio asociado a ellos.

Antes de dar solución a este problema, hay que hacer notar que el hecho de que una determinada propiedad o atributo no sea aplicable en un momento dado a una instancia de un item, puede ser, en algunos casos, consecuencia de un mal diseño de la propia BD o, simplemente, de un diseño en el que se ha tenido que omitir la consideración de este aspecto por cuestiones prácticas. Veamos algunos ejemplos.

Supongamos que queremos almacenar datos referentes a los empleados de una empresa, entre los que se encuentra el nombre del cónyuge. La pregunta que se plantea siempre para justificar la existencia de este tipo de información incompleta, es la siguiente *¿qué valor debe asignarse al atributo* `Nombre-Conyuge` *para aquellos empleados que son solteros?*. El problema que subyace aquí es, claramente, un problema de diseño, ya que se están considerando en la misma relación (vista como tabla), llamémosla `Empleados`, los datos de una entidad concreta `Empleado` (como pueden ser `Nombre`, `Edad`, `DNI`,...) junto con datos de una *Conexión o Relación entre entidades* como es en este caso, la relación `Casado-Con`, que conecta items tipo `Persona` entre sí. Obviamente, un buen diseño hubiera dividido nuestra tabla original de empleados en dos tablas distintas, una donde apareciesen exclusivamente los datos personales de los empleados, y otra en la que apareciese reflejado el nombre del empleado/a junto con el de su esposa/o y, si es el caso, con otros datos relacionados con esa conexión (como el número de hijos, por ejemplo).

Sin embargo, en otros muchos casos, la inaplicabilidad de un valor a un atributo es un hecho ineludible, como en el caso, por ejemplo, de *un ministro sin cartera*, o de *empleados sin centro fijo*, *jefes sin superiores*, etc...

En cualquier caso, como quiera que la posibilidad de encontrar valores no aplicables se considera, hoy por hoy, en la manipulación de BD, la solución más inmediata pasa por definir qué es o qué se entiende por valor *no aplicable*.

La solución que da Reiter en este sentido, no es una buena solución ni para el caso de BDR clásicas ni para el de las BDRD. Lo vemos con un ejemplo que él mismo propone

y que es análogo al que acabamos de mencionar.

Supongamos, ya en el entorno de las BD lógicas, que el predicado $Emp(p, e, c)$ denota una persona p cuyo estado civil es e y el nombre de su cónyuge es c. En este caso, si el estado civil de p es soltero, no tiene ningún sentido el tercer argumento. Otro ejemplo es el predicado $CondFisicas(n, e, a, p, c)$ que denota las características físicas de una persona n, tales como su edad (e), su altura (a), su peso (p) y el color de su pelo (c). De nuevo, si se tiene que almacenar dicha información para una persona calva, nos encontramos con que no tiene sentido. En estos casos, Reiter propone la inclusión de las siguientes fbf en la teoría:

$$\forall x,\ \neg Emp(p, soltero, x)$$

para el primer ejemplo, y

$$\forall x,\ \neg CondFisicas(n, e, a, p, x)$$

para el segundo.

Nótese, que este tratamiento de los valores nulos como no aplicables tiene graves inconvenientes. El primero de ellos es que no se permite representar la semántica de dicho valor, ni se permite manejarlo en manera alguna. Sin embargo, lo peor de esta aproximación es que las fbf anteriormente expresadas nos obligarían a eliminar de la base de datos todas aquellas tuplas en las que alguno de los valores para algún atributo sea *no aplicable*, esto es, se perdería toda la información a él asociada.

En este sentido, está claro que se hace necesario dar una aproximación distinta para este tipo de valores de manera que se superen los inconvenientes mencionados.

Intuitivamente, un valor *no aplicable** puede definirse como sigue:

Sea $D_\tau = \{d_{\tau_1}, d_{\tau_2}, \ldots, d_{\tau_n}\}$ el dominio de un tipo τ. El valor no aplicable para el dominio τ viene dado por la distribución

$$d_{\tau_1}/0 + d_{\tau_2}/0 + \ldots + d_{\tau_n}/0$$

*También llamado **indefinido**

esto es, la que da posibilidad 0 a todos los valores del dominio del tipo τ.

Esta definición es sensata y coherente con la semántica asociada al nulo como no aplicable y su representación no está en contradicción con las reglas de integridad enunciadas en la sección 3.2, en las que se especifica que los nombres de los valores indefinidos, Und_τ, son una excepción del hecho de que todo nombre de distribución debe llevar asociado la correspondiente extensión del predicado POS_τ (Regla R.6).

$$\forall(x),\ (NOP_\tau(x) \longrightarrow (=(x, Und_\tau)) \vee (\exists(y,z)\ (POS_\tau(x,y,z)))$$

Dado que cada valor no aplicable es diferente según el dominio subyacente (hecho que no se da en el modelo FPDB), denotaremos por $Und_{\tau_1}, Und_{\tau_2}, \ldots, Und_{\tau_n}$ a los valores indefinidos asociados a los tipos $\tau_1, \tau_2, \ldots, \tau_n$, respectivamente. Esto nos obliga a extender la definición 5.11 como sigue:

Definición 3.13 .

Sea $T = (A, W)$ una teoría relacional difusa como en la definición 5.11 y sea $D = \{Und_{\tau_1}, Und_{\tau_2}, \ldots, Und_{\tau_m}\}$, donde D puede ser el conjunto vacío. Cada uno de los elementos de D recibe el nombre de **valor nulo inaplicable**. *La Teoría Relacional Difusa $T \subseteq W$ es una* **Teoría Relacional Difusa Generalizada con Valores Desconocidos e Inaplicables** *si, y sólo si, satisface las siguientes propiedades:*

1. *Axioma de dominio cerrado:*

$$\forall x,\ =(x,c_1) \vee \ldots \vee =(x,c_m) \vee =(x,Unk_{\tau_1}) \vee =(x,Unk_{\tau_2}) \vee$$
$$\ldots \vee =(x,Unk_{\tau_n}) = (x,Und_{\tau_1}) \vee \ldots \vee =(x,Und_{\tau_m})$$

2. *Axioma de nombre único, que incluirá adicionalmente las fbf*

$$\neg =(Und_{\tau_i}, c_j)\ \ \forall i,\ 1 \leq i \leq n,\ \forall j,\ 1 \leq j \leq m$$
$$\neg =(Und_{\tau_i}, Und_{\tau_j})\ \ \forall i,\ 1 \leq i \leq n,\ \forall j,\ 1 \leq i < j$$

Nótese que, al contrario de lo que sucede con el nulo como desconocido, ahora sí deben incluirse todas las desigualdades para los valores indefinidos, ya que éstos no pueden coincidir, por definición, con ningún otro elemento del dominio.

3. Los axiomas de igualdad

4. Las extensiones de todos los predicados.

5. Sea C_p como en 3.11. Si

$$C_p = \{(c_1^1, \ldots, Und_{\tau_i}, \ldots, c_m^1), \ldots, (c_1^r, \ldots, c_m^r)\}$$

entonces el axioma de completitud incluye las fbf de Δ y el siguiente axioma de completitud para P:

$$\forall(x_1, x_2, \ldots, x_m),\ P(x_1, x_2, \ldots, x_m) \longrightarrow ..... \vee (= (x_1, c_1) \wedge \ldots \wedge$$

$$= (x_i, Und_{\tau_i}) \wedge \ldots \wedge = (x_m, c_m))$$

Si, por el contrario, $C_p = \emptyset$, entonces el axioma de completitud de P tiene la forma:

$$\forall\ (x_1, \ldots, x_m),\ \neg P(x_1, \ldots, x_m)$$

En el caso de definirse nuevos operadores, como $>$, $<$, $\leq$,... para un determinado dominio, será necesario incluir las restricciones correspondientes para el tratamiento del valor Und_τ. Por ejemplo, si tuviéramos que definir, en el dominio de un tipo τ, el comparador $<$, debería incluirse, entre otras, la siguiente regla:

$$<_\tau (x, y, z) \wedge (= (x, Und_\tau) \vee = (y, Und_\tau)) \longrightarrow = (z, Und_{MD})$$

Definición 3.14 .

Una **BDRD Generalizada con Valores Desconocidos e Inaplicables** *es un triple (R, T, RI) donde R y T son como en la definición 3.11, y $RI \subseteq W$ es un conjunto de restricciones de integridad adecuado.*

Teorema 3.3 .

Toda teoría relacional difusa generalizada con valores desconocidos e inaplicables es consistente.

Demostración:

El hecho de que se verifiquen todos los axiomas de nombre único de partida (junto con el resto de los axiomas), hace que, por el teorema 3.1, quede automáticamente demostrado que toda teoría relacional generalizada con valores desconocidos e inaplicables es consistente.

◇ El Nulo como Desconocimiento Total

La semántica que para nosotros (y para casi todos los modelos) tiene el valor nulo es el total desconocimiento del valor que corresponde a un atributo dentro de una tupla (equivalentemente, a un argumento dentro de un predicado). Este desconocimiento total y absoluto podría expresarse como que no se sabe, ni siquiera, si es un valor desconocido pero aplicable o un valor inaplicable.

Según este significado, la representación mas sensata y más en consonancia con las anteriores, sería:

$$Null_\tau = Unk_\tau/1 + Und_\tau/1$$

Obsérvese, que estamos definiendo una distribución de posibilidad de segundo orden, esto es, los elementos del dominio subyacente son, a su vez, distribuciones de posibilidad sobre otro dominio de nivel inferior.

Dado que esta posibilidad no se contemplaba hasta ahora en el modelo, vamos a ver qué reglas de integridad es necesario incluir para darle cabida de manera natural.

Intuitivamente, y antes de dar la descripción formal, lo que hay que hacer es obligar a que NOP_τ sea, a su vez, un tipo difuso, esto es, que los nombres de las distribuciones de posibilidad puedan formar parte de distribuciones de posibilidad. Una vez hecho esto, restringimos esta posibilidad sólo a los valores nulos, desconocidos y no aplicables*.

*Esta restricción no es necesaria, ya que podría permitirse definir distribuciones de posibilidad sobre otras distribuciones de posibilidad en cualquier dominio para cualquier valor. No lo haremos por simplicidad.

Para considerar que NOP_τ **es un tipo difuso**, será necesario incluir las siguientes reglas de integridad:

D1.- $\forall\ (x,y,z),\ POS_{NOP_\tau}(x,y,z) \longrightarrow NOP_{NOP_\tau}(x) \wedge NOP_\tau(y) \wedge MD(z) \wedge \neg = (z,0)$

D2.- $\forall x,\ NOP_{NOP\tau}(x) \longrightarrow \exists\ (y,z),\ POS_{NOP_\tau}(x,y,z)$

D3.- $\forall x,\ NOP_\tau(x) \longrightarrow POS_{NOP_\tau}(x,x,1) \wedge \neg(\exists\ (y,z),\ POS_{NOP_\tau}(x,y,z) \wedge \neg = (x,y))$

D4.- $\forall\ (x,y,z),\ NOP_{NOP_\tau}(x) \longrightarrow = (x,Null_\tau) \vee = (x,Unk_\tau) \vee = (x,Und_\tau)$

La extensión del predicado POS_{NOP_τ} quedaría como sigue:

$$POS_{NOP_\tau}(Null_\tau, Null_\tau, 1)$$

$$POS_{NOP_\tau}(Null_\tau, Unk_\tau, 1)$$

$$POS_{NOP_\tau}(Null_\tau, Und_\tau, 1)$$

$$POS_{NOP_\tau}(Unk_\tau, Unk_\tau, 1)$$

$$POS_{NOP_\tau}(Und_\tau, Und_\tau, 1)$$

y la del predicado $\cong_{NOP_\tau}$ quedaría:

$$\cong_{NOP_\tau}(Null_\tau, Null_\tau, 1)$$

$$\cong_{NOP_\tau}(Null_\tau, Unk_\tau, 1)$$

$$\cong_{NOP_\tau}(Null_\tau, Und_\tau, 1)$$

$$\cong_{NOP_\tau}(Unk_\tau, Unk_\tau, 1)$$

$$\cong_{NOP_\tau}(Und_\tau, Und_\tau, 1)$$

La teoría relacional que se construye teniendo en cuenta la incorporación de valores nulos, sería como sigue:

1. Axioma de dominio cerrado, incluyendo los nombres de los valores nulos.

2. Axioma de nombre único, que incluirá

$$\neg = (Null_{\tau_i}, c_j) \; para \; algunos \; 1 \leq i \leq n, \; 1 \leq j \leq m$$

$$\forall i, \forall j, \neg = (Null_{\tau_i}, Null_{\tau_j})$$

3. Los axiomas de igualdad

4. Las extensiones de todos los predicados.

5. Los predicados asociados al conjunto C_p como en la definición 3.11.

Como en el caso de los valores desconocidos, la única *"anomalía"* que se presenta es que no se consideran todas las desigualdades en el axioma de nombre único, lo cual no es un impedimento para que la teoría construida sea consistente.

Teorema 3.4 .

Toda teoría relacional difusa generalizada con valores desconocidos, inaplicables o nulos es consistente.

Definición 3.15 .

Llamaremos **Teoría Relacional Difusa Completa** *a una teoría relacional difusa en la que se inlcuye información disyuntiva, así como cualquiera de los tipos de información omitida considerado (desconocida, inaplicable o nula).*

Definición 3.16 .

Una **BDR Difusa Completa** *es un triple* (R, T, RI) *donde* R *es un lenguaje relacional,* T *es una teoría relacional difusa completa (según la definición anterior) y* RI *es el conjunto de reglas de integridad necesario para la correcta manipulación de los datos.*

Nota: Entre las reglas de integridad deben incluirse todas las restricciones o reglas necesarias para operar con los diferentes tipos de datos nulos.

3.6 Reformulación de Consultas

Como hemos podido ver, los elementos introducidos para el manejo de datos difusos, no impiden realizar cualquier tipo de consulta, ya sea de tipo clásico o de tipo difuso a la BD. Lo que sí ha podido observarse, es que éstas últimas resultan algo tediosas en la formulación, ya que involucran grados de cumplimiento así como atributos y comparadores difusos. Este problema podríamos formularlo de forma genérica como sigue:

Consideremos un predicado $P(-,-)$ y supongamos que el segundo atributo, τ_2, es difuso. Sea $e \in NOM_{\tau_2}$ el nombre de una distribución de posibilidad sobre el dominio de τ_2. Si formulamos una consulta precisa, que involucre el predicado P, de la forma:

$$< x/\tau_1 \mid P(x, e) >$$

sólo se tendrán en cuenta aquellos valores de x para los que se verifica $P(x, e)$, sin tenerse en cuenta aquéllos para los que se da $P(x, y)$ y $\cong_{\tau_2} (e, y, z)$ con un valor para z próximo a 1. De hecho, la única forma de incluir todas las respuestas posibles cuando se hace una consulta de este tipo, es incluyendo siempre el predicado $\cong_\tau$ así como, cuando se exija, una variable que represente los grados de cumplimiento (y eso, para cada atributo difuso involucrado).

En el primero de los casos, el planteamiento es *"crisp"*, en el sentido de que se recuperará sólo aquella información que cumpla rigurosamente las condiciones exigidas

en la consulta. En el segundo caso, la resolución de la consulta supone *rastrear* todas las posibilidades que más o menos se ajusten a los requisitos de la consulta. Esta última forma de consulta está ya, implícitamente, en la filosofía de la lógica difusa.

Dentro de esta última aproximación [123] hay también dos formas generales de formular una consulta como: *"Encontrar aquellos valores de x tales que* $P(x, e)$ *siendo e una distribución de posibilidad"*, que son:

1. $< x/\tau_1 \mid \exists\ (y, z),\ (P(x, y) \wedge \cong_{\tau_2} (e, y, z) \wedge \geq (z, \alpha) >$, donde α es un grado de cumplimiento mínimo en la comparación.
2. $< x/\tau_1,\ z/MD \mid \exists y,\ (P(x, y) \wedge \cong_{\tau_2} (e, y, z) >$

3.6.1 Elementos Adicionales

Para poder generalizar el caso mostrado en el apartado anterior, será necesario introducir algunos elementos adicionales, pertenecientes al campo de la Lógica Difusa. En particular, si nos encontramos ante una fórmula compleja que involucre los operadores $\wedge$, $\vee$ y $\neg$, tendremos que manejar las aproximaciones difusas de dichos operadores lógicos. A su vez, será también necesario incluir información acerca de cómo se combinarán los grados de cumplimiento de las condiciones bajo dichos operadores. Todo ello puede hacerse introduciendo un conjunto apropiado de predicados que representen estas operaciones. Aunque la Lógica Difusa utiliza normalmente MIN, MAX y $(1 - x)$ respectivamente, es posible generalizarlas utilizando pares duales de T-Normas y T-Conormas y la *negación fuerte* [1]. Así pues, definiremos operaciones generales sobre el dominio del tipo MD de manera que verifiquen las propiedades exigidas.

* *Definición de la Negación Fuerte*

  Introduciremos un nuevo predicado en L, llamado $NEG(-, -)$, que verifica:

  N1. $\forall x,\ (MD(x) \longrightarrow \exists y,\ MD(y) \wedge NEG(x, y))$

  N2. $\forall\ (x_1, x_2, y_1, y_2),\ (\geq (x_1, x_2) \wedge \neg = (x_1, x_2) \wedge NEG(x_1, y_1) \wedge NEG(x_2, y_2) \longrightarrow \geq (y_2, y_1) \wedge \neg = (y_1, y_2))$

  N3. $\forall\ (x, y, z),\ (NEG(x, y) \wedge NEG(y, z) \longrightarrow = (x, z))$

N4. $NEG(1,0)$

N5. $NEG(0,1)$

$*$ *Definición de un Par Dual T−Norma/T−Conorma*

Introduciremos dos nuevos predicados en L, $TN(-,-,-)$ y $TCN(-,-,-)$, que verifican:

T1. $\forall\ (x,y,z),\ (MD(x) \wedge MD(y) \wedge MD(z))$

T2. $\forall\ (x,y),\ (TN(x,1,y) \longrightarrow TN(1,x,y) \wedge = (y,x))$

T3. $\forall\ (x,y,z,v_1,v_2,w_1,w_2),\ (TN(x,y,v_1) \wedge\ TN(v_1,z,w_1) \wedge TN(y,z,v_2) \wedge$
$TN(x,v_2,w_2) \longrightarrow = (w_1,w_2))$

T4. $\forall\ (x,y,z),\ (TN(x,y,z) \longrightarrow TN(y,x,z))$

T5. $\forall\ (x_1,x_2,x_3,x_4,z_1,z_2),\ (\geq (x_3,x_1) \wedge \geq (x_4,x_2) \wedge TN(x_1,x_2,z_1) \wedge TN(x_3,x_4,z_2)$
$\longrightarrow \geq (z_2,z_1))$

T6. $\forall\ (x,y,z),\ (TCN(x,y,z) \longleftrightarrow \exists\ (v,w,t),\ (TN(v,w,t) \wedge$
$NEG(x,v) \wedge NEG(y,w) \wedge NEG(z,t)))$

Las propiedades **T1** a **T5** caracterizan TN como una función de $MD \times MD$ en MD, no creciente, conmutativa y asociativa. La propiedad **T6** garantiza estas mismas características para TCN.

Gracias a estas propiedades, podemos dar expresiones generales para las consultas difusas. Supongamos que las constantes $k_1, k_2, \ldots, k_q$ son difusas con dominios respectivos $\tau_{n+1}, \tau_{n+2}, \ldots, \tau_{n+q}$. Bajo esta hipótesis, tenemos, de nuevo, dos tipos de consultas atómicas, que expresaremos por:

1. $< x_1/\tau_1, \ldots, x_n/\tau_n \ \mid\ \exists\ (y_{n+1}, \ldots, y_{n+q}, z_{n+1}, \ldots, z_{n+q}, v_{n+r+1}, \ldots, v_m),$
$P(x_1, \ldots, x_n, y_{n+1}, \ldots, y_{n+q}, z_{n+1}, \ldots, z_{n+q}, v_{n+r+1}, \ldots, v_m))$
$\wedge \cong_{\tau_{n+1}} (k_1, y_{n+1}, z_{n+1}) \wedge \ldots \wedge \cong_{\tau_q} (k_q, y_{n+q}, z_{n+q}) \wedge$
$\geq (z_{n+1}, \alpha) \wedge \ldots \wedge \geq (z_{n+q}, \alpha)) >$

   donde α es un grado de cumplimiento exigido en la consulta.

2. $< x_1/\tau_1, \ldots, x_n/\tau_n, z/MD \ \mid$
$\exists\ (y_{n+1}, \ldots, y_{n+q}, z_{n+1}, \ldots, z_{n+q}, v_{n+r+1}, \ldots, v_m, w_1, \ldots, w_{q-1}),$

$$P(x_1, \ldots, x_n, y_{n+1}, \ldots, y_{n+q}, z_{n+1}, \ldots, z_{n+q}, v_{n+r+1}, \ldots, v_m)) \wedge$$
$$\cong_{\tau_{n+1}} (k_1, y_{n+1}, z_{n+1}) \wedge \ldots \wedge \cong_{\tau_q} (k_q, y_{n+q}, z_{n+q}) \wedge$$
$$TN(z_{n+1}, z_{n+2}, w_1) \wedge TN(w_1, z_{n+3}, w_2) \wedge \ldots \wedge TN(z_{n+q}, w_{q-1}, z)) >$$

3.6.2 Definición de Predicados Difusos

Como puede verse, una consulta medianamente compleja formulada en estos términos, puede resultar bastante complicada si no se introduce algún mecanismo adicional que la simplifique. La solución es inmediata, ya que podemos definir, gracias a la caracterización lógica con la que estamos trabajando, un conjunto de predicados *intensionales* que nos permitan simplificar dichas fórmulas. Veámoslo.

Supongamos que (L, T, IC) es una BD Lógica Difusa, donde $L = (a, W)$. Consideremos el conjunto P de los símbolos de predicados $n-$arios de A $(n > 1)$, tales que $\forall P,\ P \in$P se verifica que:

(i) P es distinto de $=$ y de $\geq$.

(ii) P no está directamente relacionado con ningún tipo difuso en particular, esto es, es distinto de POS_τ, de $\cong_\tau$ y de cualquier otro comparador definido para τ.

(iii) Entre los dominios de los atributos de P hay, al menos, un tipo difuso.

Si P es cualquier predicado del conjunto P, asumiremos, sin ninguna pérdida de generalidad, que los q primeros atributos serán los difusos. Siguiendo esta notación, para cualquier P, $P \in$P, definiremos su *predicado intensional asociado* $\underline{P}$ en los siguientes términos:

$$\forall\ (y_1, \ldots, y_n, z),\ (\exists\ (x_1, \ldots, x_n, z_1, \ldots, z_q, v_1, \ldots, v_{q-1})\ tales\ que$$
$$P(x_1, \ldots, x_n) \wedge \cong_{\tau_1} (x_1, y_1, z_1) \wedge \ldots \wedge$$
$$\cong_{\tau_q} (x_q, y_q, z_q) \wedge \ldots \wedge = (x_{q+1}, y_{q+1}) \wedge \ldots \wedge = (x_n, y_n) \wedge$$
$$TN(z_1, z_2, v_1) \wedge TN(z_3, v_1, v_2) \wedge \ldots \wedge TN(z_q, v_{q-2}, z)$$

$\longrightarrow \underline{P}(y_1, \ldots, y_n, z))$

Aunque la definición de $\underline{P}$ parece complicada, su significado es muy intuitivo, ya que se obtiene como generalización del Principio de Sustitución de Términos Iguales de Leibnitz, salvo que para los atributos difusos se utiliza el operador $\cong$ en lugar del $=$. Con esta definición, $\underline{P}$ puede considerarse como el predicado difuso asociado a P. Llamaremos $\underline{\mathrm{P}}$ al conjunto de todos predicados intensionales construidos a partir de P de esta manera. Obviamente, todos los símbolos de $\underline{\mathrm{P}}$ deberán ser incluídos en A. Con esta nueva definición, los dos posibles tipos de consulta anteriormente mencionadas pasan a expresarse como sigue:

1. $< x_1/\tau_1, \ldots, x_n/\tau_n \mid (\underline{P}(x_1, \ldots, x_n, k_{n+1}, \ldots, k_r, z) \wedge \geq (z, \alpha)) >$, donde α es el grado de cumplimiento.
2. $< x_1/\tau_1, \ldots, x_n/\tau_n, z/MD \mid (\underline{P}(x_1, \ldots, x_n, k_{n+1}, \ldots, k_r, z) >$

que, como puede verse, es una simplificación considerable respecto a las anteriores expresiones.

3.6.3 Expresión de Consultas no Atómicas

Vamos a ver cómo quedarían representadas aquellas consultas que involucran predicados difusos y operadores lógicos difusos. Por simplicidad, adoptaremos la siguiente notación para los predicados difusos: $\underline{P}(\overline{x}, \overline{e}, v)$, donde $\overline{x} = x_1, x_2, \ldots, x_q$ variables cuyos dominios respectivos son $\overline{\tau} = \tau_1, \tau_2, \ldots, \tau_q$, $\overline{e} = e_1, e_2, \ldots, e_r$ son constantes y v es la variable definida sobre el dominio de MD y que caracteriza a $\underline{P}$ como difuso. Obviamente, la ariedad de $\underline{P}$ será $q + r + 1$ donde r puede ser, eventualmente, 0. Con esta notación, podríamos expresar fácilmente consultas compuestas tales como:

- *Encontrar los valores de* $\overline{x}$ *tales que* $NOT(\underline{P}(\overline{x}, \overline{e}, v))$.
  - $< \overline{x}/\overline{\tau} \mid \exists z, (\underline{P}(\overline{x}, \overline{e}, z) \wedge NEG(z, \alpha)) >$ si se exige un grado de cumplimiento α.
  - $< \overline{x}/\overline{\tau}, v/MD \mid \exists z, (\underline{P}(\overline{x}, \overline{e}, z) \wedge NEG(z, v) >$

- *Encontrar los valores de* $\overline{x}, \overline{y}$ *tales que* $\underline{P}(\overline{x}, \overline{e}, v)\ AND\ \underline{Q}(\overline{y}, \overline{k}, w)$

  - $< \overline{x}/\overline{\tau}, \overline{y}/\overline{\tau} \mid \exists\,(z,w,t)\ (\underline{P}(\overline{x}, \overline{e}, z)\ AND\ \underline{Q}(\overline{y}, \overline{k}, w) \wedge TN(z,w,t) \wedge \geq (t, \alpha)) >$ para consultas con umbral.
  - $< \overline{x}/\overline{\tau}, \overline{y}/\overline{\tau}, v/MD \mid \exists\,(z,w)\ (\underline{P}(\overline{x}, \overline{e}, z)\ AND\ \underline{Q}(\overline{y}, \overline{k}, w) \wedge TN(z,w,v)) >$

- *Encontrar los valores de* $\overline{x}, \overline{y}$ *tales que* $\underline{P}(\overline{x}, \overline{e}, v)\ OR\ \underline{Q}(\overline{y}, \overline{k}, w)$

  - $< \overline{x}/\overline{\tau}, \overline{y}/\overline{\tau} \mid \exists\,(z,w,t)\ (\underline{P}(\overline{x}, \overline{e}, z)\ OR\ \underline{Q}(\overline{y}, \overline{k}, w)) \wedge TCN(z,w,t) \wedge \geq (t, \alpha) >$ para consultas con umbral.
  - $< \overline{x}/\overline{\tau}, \overline{y}/\overline{\tau}, v/MD \mid \exists\,(z,w)\ (\underline{P}(\overline{x}, \overline{e}, z)\ OR\ \underline{Q}(\overline{y}, \overline{k}, w)) \wedge TCN(z,w,v) >$

3.6.4 Ejemplos

Consideremos la BD lógica difusa que se describe en la sección 3.2.3. Sobre ella, realizaremos las siguientes consultas:

1. *"Encontrar los nombres de los alumnos que no sean muy jóvenes".*

   $< x/Nombre, w/MD \mid \exists\,(y,t,v),\ \underline{Alumno}(x, Muy\ Joven, y, t, v) \wedge NEG(v,w) >$

2. *"Encontrar los nombres de los alumnos que sean jóvenes o muy jóvenes con grado mínimo 0.8".*

   $$< x/Nombre \mid \exists\,(y,t,v,w),\ (\underline{Alumno}(x, Joven, y, t, v) \vee$$
   $$\underline{Alumno}(x, Muy\ Joven, y, t, w)) \wedge TCN(v, w, 0.8) >$$

3. *"Encontrar los nombres de los alumnos que sean jóvenes y tengan un comportamiento bueno ".*

   $$< x/Nombre, v/MD \mid \exists y,\ \underline{Alumno}(x, Joven, y, Bueno, v) >$$

◇ Definición General de Predicado Difuso Intensional

En la sección anterior, hemos visto cómo es posible definir un predicado difuso inducido por predicados o hechos que se encuentran en una BD. En esta sección extendemos el concepto de predicado difuso intensional, de manera que éste puede estar inducido, en general, por cualquier conjunto de reglas o predicados, esto es, una vez definido de una manera un tanto restrictiva (consideramos sólo un atributo difuso) el concepto de *predicado difuso intensional*, pasamos a dar una definición formal del mismo, ya con carácter general.

Definición 3.17 .

Sea (L, T, IC) *una base de datos lógica difusa, donde* $L = (A, W)$, *sea* $\underline{Q}_{11}$, ..., $\underline{Q}_{1(r1)}$, ..., $\underline{Q}_{n1}$, ..., $\underline{Q}_{n(rn)}$ *un conjunto de símbolos de comparadores difusos y sea* $\underline{S}_{11}$, ..., $\underline{S}_{1(q1)}$, ..., $\underline{S}_{n1}$, ..., $\underline{S}_{n(qn)}$ *un conjunto de símbolos de predicados no difusos. En esta situación, diremos que el símbolo de predicado* $\underline{P}$ *es un* **predicado difuso intensional** *si, y sólo si:*

$$\exists\ (\underline{Q}_{11}, \ldots, \underline{Q}_{1(r1)}, \ldots, \underline{Q}_{n1}, \ldots, \underline{Q}_{n(rn)})$$

y

$$(\underline{S}_{11}, \ldots, \underline{S}_{1(q1)}, \ldots, \underline{S}_{n1}, \ldots, \underline{S}_{n(qn)})$$

tales que se encuentren en T *las fórmulas siguientes:*

$$\forall\ (\overline{x}^1, v_1),\ \exists\ (\overline{y}^1, v_{11}, \ldots, v_{1(r1)}, z_1, \ldots, z_{r1-1})$$

$$\underline{Q}_{11}(\overline{x}^{11}, \overline{e}^{11}, v_{11}) \wedge \ldots \wedge \underline{Q}_{1(r1)}(\overline{x}^{1(r1)}, \overline{e}^{1(r1)}, v_{1(r1)}) \wedge$$

$$TN(v_{11}, v_{12}, z_1) \wedge TN(v_{13}, z_1, z_2) \wedge \ldots \wedge TN(v_{1(r1)}, z_{r1-1}, v_1) \wedge \underline{S}_{11}(\overline{t}^{11}, \overline{k}^{11}) \wedge \ldots \wedge$$

$$\underline{S}_{1(q1)}(\overline{t}^{1(q1)}, \overline{k}^{1(q1)}) \longrightarrow \underline{P}(\overline{x}^1, v_1)$$

.....

$$\ldots\ldots\ldots$$

$$\forall\ (\overline{x}^n, v_n),\ \exists\ (\overline{y}^n, v_{n1}, \ldots, v_{n(r1)}, z_1, \ldots, z_{rn-1})$$

$$\underline{Q}_{n1}(\overline{x}^{n1}, \overline{e}^{n1}, v_{n1}) \wedge \ldots \wedge \underline{Q}_{n(rn)}(\overline{x}^{n(rn)}, \overline{e}^{n(rn)}, v_{n(rn)}) \wedge$$

$$TN(v_{n1}, v_{n2}, z_1) \wedge TN(v_{n3}, z_1, z_2) \wedge \ldots \wedge$$

$$TN(v_{n(rn)}, z_{rn-1}, v_n) \wedge \underline{S}_{n1}(\overline{t}^{n1}, \overline{k}^{n1}) \wedge \ldots \wedge$$

$$\underline{S}_{n(qn)}(\overline{t}^{n(qn)}, \overline{k}^{n(qn)}) \longrightarrow \underline{P}(\overline{x}^n, v_n)$$

donde,

- $\forall\ (i,j,k),\ \mid\ i = 1, \ldots, n,\ j = 1, \ldots, ri\ y\ k = 1, \ldots, qi$
- $\exists m,\ \mid\ \overline{x}^i = x^i_1 \ldots x^i_m\ y\ \forall l = 1, \ldots, m$ *el dominio de* x^i_l *es* τ_l. *Esto es, todos los vectores* $\overline{x}^i$ *tienen el mismo número de variables y con los mismos dominios.*
- *Las variables del vector* $\overline{x}^{ij}$ *están en* $\overline{x}^i$ *o en* $\overline{y}^i$.
- *Las variables del vector* $\overline{x}^{ik}$ *están en* $\overline{x}^i$ *o en* $\overline{y}^i$.

Con esta definición, es posible establecer la extensión de un predicado difuso intensional como sigue:

Definición 3.18 .

Sea $\overline{c} = c_1, c_2, \ldots, c_n$ *un vector de constantes y* α *tal que* $T \vdash MD(\alpha)$. *El par* $(\overline{c}, \alpha)$ *pertenece a la extensión de* $\underline{P}$ *si, y sólo si:*

- $\forall q,\ q = k+1, \ldots, m,\ T \vdash \tau_q(c_q)$
- $T \vdash \underline{P}(\overline{c}, \alpha)$
- $\forall \beta,\ \mid\ T \vdash \underline{P}(\overline{c}, \beta)$ *se cumple* $T \vdash \geq (\alpha, \beta)$

Una vez establecida esta definición, una predicado difuso intensional tiene el mismo significado que cualquier otro predicado difuso. Así pues podemos realizar la siguiente definición:

Definición 3.19 .

Llamaremos $\underline{F}$ *al* **conjunto de predicados difusos de** A, *que contendrá los siguientes símbolos:*

$$\underline{F} = \underline{P} \cup \{POS_\tau \mid \forall \tau \ tipo\ difuso\} \cup \{\Theta_\tau \mid \forall \Theta \ comparador\ difuso\}$$

donde $\underline{F}$ representa el conjunto de símbolos de predicado que tienen una relación difusa como interpretación.

3.7 Conclusiones

A lo largo de este capítulo, se han introducido los elementos esenciales para el manejo de una BDRD desde la perspectiva de la lógica de primer orden. En primer lugar, se ha aproximado dicha BD desde el punto de vista de la Teoría de Modelos, en la que la BD de partida se considera una interpretación para un lenguaje relacional especial. Posteriormente, se ha aproximado desde el punto de vista de la Teoría de Pruebas, en la que la BD se considera una teoría de primer orden. Igual que para el caso clásico, se demuestra que ambas aproximaciones son equivalentes.

Tras esta aproximación, vemos cómo se identifican de forma natural, los principales modelos de BDD de la literatura como casos particulares de nuestra definición, tanto en la estructura de los datos como en la manipulación de los mismos.

De las dos aproximaciones iniciales, nos hemos decantado (la elección está sobradamente justificada) por la de la teoría de pruebas, ya que nos ofrece un marco mucho más flexible para representar tanto información disyuntiva como información omitida en sus distintas variantes. Esta aproximación nos permite, a su vez, la definición de predicados difusos de forma intensiva (sin necesidad de darlos por extensión) y utilizarlos en las consultas, haciendo que éstas sean mucho más sencillas y compactas, casi tanto como las del caso clásico.

En resumen, hemos demostrado que es posible definir una BD Lógica Difusa como una estructura lógica que no difiere, en lo esencial, de la utilizada para el caso clásico.

Todo gracias al poder expresivo de la lógica de primer orden, que nos has permitido manejar consultas difusas de manera natural y directa gracias a la definición de elementos de la lógica difusa y de los predicados difusos como predicados. Estos resultados son de gran importancia, ya que permiten utilizar las herramientas computacionales existentes para las BD clásicas para la manipulación de las BD difusas. El único punto que nos queda por resolver es el de la definición de predicados difusos intensionales que llevan asociado un grado de incertidumbre. Trataremos de resolver este problema en los capítulos siguientes.

Capítulo 4

Inferencia a partir de una BD Difusa mediante Reglas Imprecisas

El siguiente capítulo trata de llevar a cabo la definición y representación de reglas tanto difusas como precisas, de manera que se puedan definir conceptos de forma intensiva a partir de los hechos almacenados en una Base de Datos Lógica Difusa.

Lo más interesante de nuestro trabajo, aparte de permitir manejar datos imprecisos e inciertos, estriba en la posibilidad de poder modelizar las propiedades que describen las reglas desde distintos puntos de vista. Esto es, permite utilizar, simultáneamente, tipos distintos de reglas: reglas de tipo posibilístico (el grado de incertidumbre asociado a la regla es un valor de posibilidad), reglas de tipo probabilístico (el grado de incertidumbre asociado es una probabilidad) y reglas clásicas. Gracias a dichas reglas va a ser posible, adicionalmente, *ampliar virtualmente* las relaciones de la BD definiendo toda una gama de valores de un dominio sobre el cual podrán definirse atributos difusos.

El capítulo se presenta estructurado de la siguiente forma. En la primera parte, se dan las características más importantes del esquema de deducción que proponemos, dando, previamente, una interpretación concreta tanto a las reglas cualificadas posibilísticamente como a las probabilísticas. A continuación, se lleva a cabo la descripción formal de dicho esquema de deducción, sobre el que se destacan las propiedades más relevantes. En una sección posterior, se muestra cómo, gracias al mecanismo de deducción empleado, se puede ampliar virtualmente una BD, utilizando un conjunto de

reglas que verifican unas propiedades y definiendo lo que hemos llamado *atributos virtuales difusos*. Por último, todos los puntos anteriormente señalados son tratados desde la perspectiva de las BDLD, para lo cual se hace necesario añadir a nuestro lenguaje algunos símbolos y a nuestra teoría, algunas restricciones adicionales.

4.1 Introducción

Cuando se lleva a cabo la representación lógica de una Base de Datos (llamémosla clásica), la principal ventaja que puede encontrarse es la posibilidad de definir conceptos sin necesidad de representarlos de manera extensiva sino de forma intensiva, esto es, mediante reglas. Gracias a las reglas, pueden llevarse a cabo deducciones que permiten obtener información adicional a la que está explícitamente almacenada en la BD (en este caso, en forma de hechos).

En este sentido, existen ya muchos modelos en la literatura que, utilizando técnicas de deducción automática, permiten incluso derivar respuestas que son conjuntos de fórmulas. Estas deducciones hacen uso tanto de la información de la BD como de una base de datos adicional que contiene las reglas, y que suele recibir el nombre de **Base de Reglas** (BR). La información que se almacena en la BR es independiente del estado actual de los datos, ya que esta última información se pone de manifiesto en la información que contiene en un momento dado la BD. Las reglas representan, pues, propiedades invariantes del universo del discurso en consideración.

Bajo este esquema común de deducción, existen distintas variantes, que se diferencian, fundamentalmente, en la forma en que resuelven y responden a las consultas (ver [55], [56], [41], [43], [103] y [43]).

En todos estos casos, tanto la información de partida como la información que se deriva, se considera precisa (salvo la información disyuntiva y los valores nulos) y completamente cierta.

Sin embargo, cuando se maneja información imprecisa en la BD, aparecen una serie de inconvenientes adicionales derivados de la naturaleza imprecisa de los hechos de partida y, en muchos casos, también de las reglas.

Hasta ahora, la mayoría de los sistemas difusos basados en reglas que aparecen en la literatura tratan este problema desde un punto de vista muy restrictivo, ya que consideran un sólo tipo de regla difusa en el proceso de inferencia, esto es, un tipo determinado de imprecisión en la misma. Sin embargo, la imprecisión en las reglas puede tener distinto significado y medida, lo que hace que sea de gran importancia el desarrollo de sistemas capaces de manejar diferentes tipos de reglas imprecisas durante el mismo proceso de inferencia. Además, el enfoque que se da en dichos sistemas es también diferente, ya que suele tratarse de deducir a partir de unos *pocos hechos* y *muchas reglas*, mientras que, en nuestro caso, se trata de obtener la extensión de un predicado que está definido por *algunas reglas* en base a una *gran base de hechos* que es la BD.

El problema que nos planteamos es, pues, el deducir y definir nueva información (no contemplada hasta ahora en la BD) a partir de la contenida explícitamente en dicha Base de Datos. Para ello, se incorporará al sistema los elementos necesarios para poder definir y utilizar diferentes tipos de reglas.

En adelante, supondremos la existencia de una Base de Datos Lógica Difusa. Los registros de dicha BD, representados ahora por hechos (como hemos visto en capítulos anteriores), pueden llevar como argumentos tanto valores precisos como valores difusos.

4.2 Características Generales del Modelo de Deducción

Antes de tratar el problema de la deducción en el marco de las BDLD, comentamos brevemente el problema general de la deducción con proposiciones imprecisas e inciertas.

Zadeh, en [133], clasifica las proposiciones que pueden manejarse en una Base de Conocimiento en cuatro categorías, que son:

1. Proposiciones incondicionales sin cualificar.

   X es A

2. Proposiciones incondicionales cualificadas.

X es A es α

3. Proposiciones condicionales sin cualificar.

Si X es A entonces Y es B

4. Proposiciones condicionales cualificadas.

Si X es A entonces Y es B es α

Hasta ahora hemos manejado los tipos 1 y 2, que se identifican con los hechos almacenados en la BD y con las respuestas a las consultas que se realizan sobre los mismos, respectivamente.

Los tipos de proposiciones 3 y 4 son los que vamos a considerar a continuación. Si bien, las proposiciones de tipo 3, que son las reglas clásicas, pueden ser directamente aplicadas a los hechos almacenados en la BD, con respecto a las de tipo 4, habría que analizar la naturaleza de la medida de incertidumbre que se ha asociado a la misma y, en función de la interpretación que se le dé, utilizar uno u otro mecanismo de propagación de la misma.

Hasta hace poco tiempo, ha existido una aceptación universal de la creencia en que la información que se maneja es, intrínsecamente, de naturaleza estadística. Por ello, se han utilizado, para su manejo, los métodos que ofrece la teoría de la probabilidad. Sin embargo, existen muchos problemas que escapan de su ámbito (ver [48]). En los últimos años, han surgido numerosas aplicaciones en las que cobra gran importancia, no sólo la manipulación de los datos, sino también y de forma predominante su *significado*.

Cuando nuestro interés se centra en el significado de la información, más que en su medida, el marco de trabajo en el que nos movemos es *posibilístico*, más que probabilístico y las herramientas propias serán las que ofrece la teoría de la posibilidad. En este sentido, la teoría clásica de la probabilidad es insuficiente para expresar la gran variedad de clases de incertidumbre que uno puede encontrarse.

Así pues, en lo referente a información imprecisa, nos encontramos en la vida real con dos tipos distintos de incertidumbre, la aleatoriedad y la ambigüedad. Desde

que Zadeh definió el concepto de conjunto difuso, la relación entre probabilidad y posibilidad se ha discutido mucho. La comparación entre ambas es compleja por dos motivos, principalmente: por un lado existe cierta similitud formal entre la teoría de la posibilidad y la de la probabilidad pero, por otro lado, la probabilidad sólo nos sirve para expresar incertidumbre en el pasado. Aún así, ambas están cuantitativamente relacionadas, esto es, un grado alto de probabilidad siempre implica un grado alto de posibilidad, pero no al revés. Sin embargo, si un suceso es imposible, está obligado a ser improbable. Esta conexión entre probabilidad y posibilidad se establece en lo que se llama *Principio de Consistencia Probabilidad/Posibilidad* [132], de gran importancia en la toma de decisiones bajo incertidumbre.

Lo que está claro es que el conocimiento humano está basado tanto en apreciaciones de tipo probabilístico (basadas en la repetición de un fenómeno) como en apreciaciones subjetivas o particulares, lo que hace necesario trabajar, simultáneamente, con probabilidad y posibilidad si se desea modelizar, de manera más o menos fiel, este conocimiento. Por ejemplo, en medicina es bastante común encontrar afirmaciones del tipo:

"El paracetamol produce vómitos en 2 de cada 1000 casos "

que es, claramente, probabilística, ya que está fundamentada en un estudio previo de pacientes a los que se ha sumistrado tal medicamento. Sin embargo, una afirmación del tipo:

"Una persona es atractiva si tiene un aspecto agradable, una conversación inteligente y sentido del humor"

es una apreciación más o menos subjetiva sobre el atractivo de las personas.

Por ello, trataremos de dar cabida en nuestro modelo a ambos tipos de incertidumbre, manipulándolas conjuntamente.

4.2.1 Interpretación de Proposiciones Condicionales en el Marco de las BDLD

El esquema de proposiciones condicionales dado por Zadeh, ha sido interpretado de numerosas formas por los autores de la materia, en función de la aplicación que se le ha dado. En estas interpretaciones se consideran diferentes *tipos* de reglas (como condicionamiento [52], como implicación material [3] [7], como disparadores para ejecutar unas acciones [115],...) así como diferentes medidas de la incertidumbre asociada a las mismas (probabilidad, posibilidad, necesidad, valor de verdad, valor de certeza,...). Aunque todas ellas son aproximaciones válidas para modelizar los problemas concretos para los que se plantean, ninguna de ellas resulta del todo válida en el entorno en el que nos movemos, esto es, en la resolución del problema de cómo obtener información adicional a la almacenada en la BD utilizando los atributos en ella definidos. Ahora veremos por qué.

Supongamos una BDLD en la que se considera el siguiente conjunto de atributos (de la misma o de diferentes relaciones) $\{A_1, A_2, \ldots, A_n\}$ y supongamos que, a partir de ellos, se desea obtener cierta información acerca de una propiedad B. En el caso más sencillo, este problema podría modelizarse por medio de una regla (por el momento clásica) del tipo

$$Si\ A\ entonces\ B$$

o bien

$$A \longrightarrow B$$

donde A y B son, en general, conjuntos difusos de universos U y V, respectivamente. En esta situación, la *regla composicional de inferencia* [49] [77] ofrece un marco ideal para derivar, ante cualquier información relativa o parecida a A, llámese A^*, información relativa a B, permitiendo incluso construir el difuso B^* asociado a A^*, como respuesta.

Sin embargo, en nuestro caso ésta no deja de ser una situación ideal, y para verlo, piense en el siguiente ejemplo. Supóngase una BD como la del ejemplo del capítulo 5

y supóngase que se desea averiguar, en base a los atributos de dicha BD, cuáles son los mejores alumnos. Para ello, habrá que definir primero la propiedad *ser un buen alumno* en base a la información que se tiene y, posteriormente, verificar qué alumnos cumplen dicha propiedad y hasta qué punto. Supongamos que la regla que **define** dicho concepto es la siguiente:

"Un alumno se considera bueno en una asignatura si tiene la edad adecuada al curso que corresponde a dicha asignatura, si muestra una actitud positiva y si la calificación es de notable como mínimo".

A diferencia del primer ejemplo, aquí aparecen una serie de complicaciones derivadas de dos factores, fundamentalmente:

- El hecho de que *los atributos involucrados en el antecedente no tienen por qué ser todos de la misma naturaleza* (obsérvese que, mientras la **edad** es representable mediante un difuso, la **actitud** tiene un dominio discreto no numérico).
- *Resulta complejo, cuando no imposible, determinar cuál es el dominio básico de la propiedad definida* o consecuente, el cual está únicamente caracterizado por un conjunto de reglas y otro de etiquetas lingüísticas.

El primero de los inconvenientes mencionados no nos permite utilizar directamente la aritmética de conjuntos difusos, ya que no se trabaja sobre el mismo dominio subyacente, y el segundo, no nos permite *construir* el resultado. Ante estos dos grandes inconvenientes, no resulta nada fácil (a veces es imposible) *, construir un B^* asociado a una determinada premisa.

La complejidad se incrementaría enormemente si, además, cualquiera de las propiedades que se especifica en la premisa se hubiera definido, a su vez, en términos de otros atributos por medio de otra regla, situación ésta bastante deseable, por otra parte.

Todo ello nos lleva a adoptar un modelo en el que no sea relevante la forma que adopten los valores de los atributos involucrados en la definición, sino tan sólo el cumplimiento o no cumplimiento de los valores exigidos como pre-requisitos en la misma. Este problema lo tratamos a continuación, centrándonos primero en reglas

*En caso de poderse construir resultaría altamente costoso

posibilísticas, para pasar, posteriormente, a tratar el caso en el que la regla lleva asociada una probabilidad.

4.2.2 Proposiciones Condicionales Cualificadas Posibilísticamente

◇ Planteamiento Inicial del Problema

Consideraremos, en primer lugar, las reglas clásicas, en las que el valor de posibilidad asociado a la regla es 1 (así como su valor de probabilidad).

El problema que vamos a plantear puede esquematizarse, inicialmente, en los siguientes términos:

$$\frac{\begin{array}{l} Si\ X\ es\ A\ entonces\ Y\ es\ B \\ X\ es\ A^* \end{array}}{Y\ es\ B\ es\ \nu}$$

donde X e Y son variables de universos de referencia U_1 y U_2, respectivamente y A, B y A^* son, en su versión más general, distribuciones de posibilidad sobre los correspondientes dominios, que actúan como restricciones difusas sobre las variables [132].

Así pues, partiendo de un hecho de la BD que expresamos como $X\ es\ A^*$ y de una relación entre el valor de Y dado que conocemos el de X, trataremos de propagar información relativa a X hacia Y.

Según este esquema de inferencia, la solución obtenida es un valor asociado a una propiedad B, cuyo significado veremos en adelante, y que nos dará una idea de hasta qué punto un determinado individuo cumple la propiedad B. Esto es, podremos conocer cuáles son los *individuos* de la BD que verifican la propiedad B y con qué intensidad.

Informalmente hablando, este esquema pretende resolver un problema muy común en el marco de las BD difusas, que es el siguiente: si partimos de una regla *"Si X es A entonces Y es B"* y tenemos un hecho en la BD que se interpreta como *"X es*

parecido a A en un cierto grado", ¿hasta qué punto se parece Y a B?. Obsérvese que el problema es el mismo que el de consultar la BD (donde hay que considerar un predicado de igualdad *flexible* con los datos difusos), con la salvedad de que ahora la propiedad exigida no se obtiene directamente como una consulta a la BD de partida, sino que viene definida de forma intensiva por medio de una regla. Por consiguiente, el valor de compatibilidad de la precondición en relación con los datos que se hallan almacenados en la BD, se obtendrá por un proceso de *"acoplamiento"* (matching) igual al utilizado para realizar las consultas.

La ventaja principal de este esquema es que el proceso de inferencia es independiente de la forma que adopten las distribuciones de posibilidad involucradas, ya que lo único que se hace es transformar el grado de compatibilidad entre el hecho que aparece en la premisa y el hecho que aparece en la BD, en el grado de compatibilidad que existe entre la conclusión de la regla y el hecho inferido (que será un difuso B^* desconocido en principio).

A estas proposiciones las denominaremos **reglas definitorias**, en el sentido de que son el único medio a nuestro alcance de definir conceptos de forma intensiva.

Partiendo de un conjunto de hechos almacenados en la BD, y de un conjunto de reglas, exponemos, a continuación, como se lleva a cabo el proceso de deducción.

La idea general que subyace en el mecanismo de inferencia es que cuánto más parecido haya entre el antecedente de la regla y el hecho considerado, mayor deberá ser también el parecido entre el consecuente de la regla y el hecho inferido.

Así pues, para llevar a cabo dicho proceso necesitaremos, en primer lugar, una relación que llamaremos de *compatibilidad* o *igualdad difusa*, para poder determinar el parecido o la consonancia existente entre dos propiedades dadas como conjuntos difusos (en general, distribuciones de posibilidad). Dicha relación es de la forma:

$$comp : \widetilde{\mathcal{P}}\ (U) \times \widetilde{\mathcal{P}}\ (U) \longrightarrow [0,1]$$

$$comp(A,B) = Sup_x\{Min\{\mu_A(x), \mu_B(x)\}\}$$

donde $\widetilde{\mathcal{P}}\ (U)$ es cualquier elemento de las partes difusas del dominio U y la compati-

bilidad viene dada en términos posibilísticos. Esta relación debe verificar las reglas de integridad R-8 a R-11 del operador $\cong_\tau$ definido en el capítulo 3.

Teniendo en cuenta que la medida de compatibilidad usada es posibilística, a partir de una regla $A \longrightarrow B$ y de un hecho de la BD, llamémosle A^*, cuya compatibilidad con A (posibilidad condicionada de A dado A^*) es α, podemos obtener un valor de posibilidad asociado a B (según la expresión 2.27).

Obsérvese que, aunque no conocemos B^*, lo que sí sabemos es que B^* es *compatible* con B en grado $\nu \geq TN(\alpha, 1)$, o lo que en nuestro caso es igual, que B^* puede ser cualquiera de los valores del dominio V de B cuya compatibilidad con él sea superior o igual a ν. Este resultado puede verse gráficamente para conjuntos difusos en la figura 4.1.

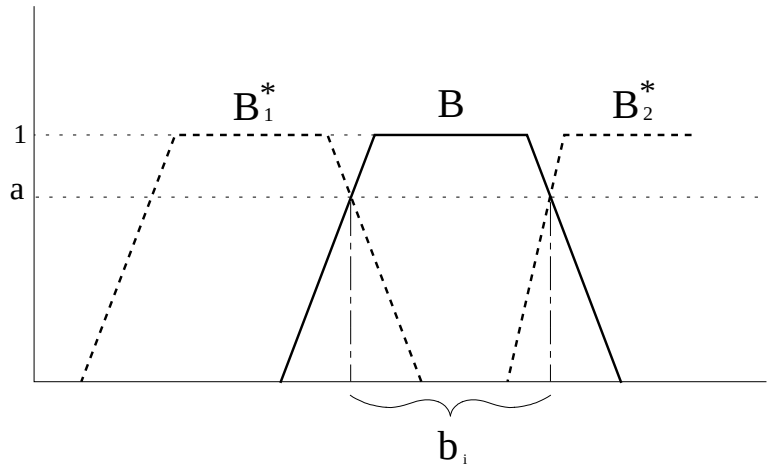


Figura 4.1. Posibles valores para el resultado B con grado a.

Nótese que, en este ejemplo, B^* puede ser cualquiera de los difusos del dominio cuya compatibilidad con B sea mayor o igual a ν (B_i^*) junto con los elementos del ν-corte de B (b_i) .

Supóngase ahora que se desea obtener información acerca de una propiedad C a partir de los datos almacenados en la BD, aún siendo éstos insuficientes para obtener una especificación completa de C, esto es, se desconocen algunos de los elementos que definen a C pero se conoce lo más relevante. En este caso, no se puede dar al resultado obtenido la misma validez o fiabilidad que a los resultados que se han obtenido a partir de reglas o definiciones completas, dado que se ha partido de información incompleta.

En este caso, deberá asignarse a la regla un grado de imprecisión que indique su validez en términos de la consonancia existente entre el objeto definido y la información utilizada para definirlo o, análogamente, que nos indique hasta qué punto es posible definir C con la información que se tiene. Esta situación puede describirse mediante el esquema siguiente:

$$\frac{\begin{array}{l}(Si\ X\ es\ A\ entonces\ Y\ es\ B)\ es\ \beta \\ \quad X\ es\ A^*\end{array}}{Y\ es\ B\ es\ \nu}$$

Este grado de fiabilidad o validez vendrá representado por un número real del intervalo $[0,1]$ y se interpretará en términos posibilísticos. Así pues, siguiendo el mismo mecanismo de propagación que el utilizado para las reglas completas, ν quedará acotado por:

$$\nu \geq TN(\alpha, \beta)$$

siendo $\alpha = comp(A, A^*)$ y β el grado de validez de la definición.

◇ Descripción Formal del Proceso de Deducción

Todas las consideraciones hechas hasta ahora tanto para reglas completas (clásicas) como para reglas imprecisas, las representaremos formalmente de la siguiente manera:

Sean $A \in \widetilde{\mathcal{P}}(U)$ y $B \in \widetilde{\mathcal{P}}(V)$ dos distribuciones de posibilidad sobre dominios U y V, respectivamente, sobre los que está definida la *regla definitoria* $(A \longrightarrow B)\ \beta$. En esta situación, la regla puede verse como una función

$$\mathcal{F} : \widetilde{\mathcal{P}}\,(\mathcal{U}) \longrightarrow \mathcal{P}(\widetilde{\mathcal{P}}\,(\mathcal{V}))$$

verificando que:

$$\mathcal{F}(A^*) = \{B_1^*, B_2^*, \ldots, B_n^*\}, \forall A^*,\ A^* \in\ \{\widetilde{\mathcal{P}}\,(U) - \{Unk_U, Und_U\}\},$$
$$\mathcal{F}(Unk_U) = Unk_V$$
$$\mathcal{F}(Und_U) = Und_V$$

donde $B_i^* \in \widetilde{\mathcal{P}}\,(V)$ y verifica que $comp(B_i^*, B) \geq\ TN(\alpha, \beta)$, siendo $comp(A, A^*) = \alpha$.

En general, si A es de la forma $A_1 \circ A_2 \circ \ldots \circ A_n$, donde $\circ$ es el operador $\vee$ o el operador $\wedge$, $\mathcal{F}$ vendrá definida de la forma:

$$\mathcal{F} :\ U_1 \times U_2 \times \ldots \times U_n \longrightarrow V$$

donde

$$comp(A_1 \wedge A_2) = TN(comp(A_1, A_1^*), comp(A_2, A_2^*))$$

$$comp(A_1 \vee A_2) = TCN(comp(A_1, A_1^*), comp(A_2, A_2^*))$$

y adoptaremos las siguientes operaciones para información omitida:

$$comp(Unk_U \wedge A) = Unk_{MD}$$

$$comp(Unk_U \vee A) = comp(A, A^*)$$

$$comp(Und_U \wedge A) = Und_{MD}$$

$$comp(Und_U \vee A) = comp(A, A^*)$$

Llamaremos $\mathcal{B}_\nu^*$ al conjunto $\{B_1^*, B_2^*, \ldots, B_n^*\}$. Intuitivamente, este conjunto estará formado por todos aquellos valores del dominio V cuya compatibilidad con B sea mayor o igual ν.

◇ Propiedades

Propiedad 1.-

$$Sean\ \beta\ y\ \beta_i\ dos\ cualificaciones\ posibilisticas\ de\ (A \longrightarrow B)\ \mid\ \beta_i < \beta.$$

$$Entonces,\ dado\ A^*,\ se\ cumple\ \mathcal{B}^*_\beta \subseteq \mathcal{B}^*_{\beta_i}$$

Nótese que para valores asociados a las reglas $\beta_i < \beta$, los conjuntos obtenidos a partir de la regla, para las mismas entradas, verifican que $\mathcal{B}^*_\beta \subseteq \mathcal{B}^*_{\beta_i}$, esto es, cuanto menor es el grado de posibilidad asociado a la regla, mayor es el abanico de posibilidades obtenido como salida. Este resultado es muy natural ya que conforme peor es una regla definitoria, mayor es la incertidumbre que produce su aplicación. Igual sucede cuando la compatibilidad entre antecendentes disminuye. Por lo general, y para garantizar resultados sensatos, se exigirá un umbral de compatibilidad mínimo entre el antecedente de la regla y el hecho a considerar, que dependerá de la importancia en sí de la regla y de lo que considere oportuno el experto que las define.

Propiedad 2.-

$$\forall A^*, A^* \subseteq A\ y\ A \longrightarrow B, \mathcal{F}(A^*) = Ker(B) \cup \mathcal{B}^*_1$$

donde $Ker(B)$ es el núcleo de B y $\mathcal{B}^*_1$ son los difusos compatibles con B en grado 1.

Informalmente, si partimos de un A^* tal que $A^* \subseteq A$, el resultado es el conjunto formado por los elementos del núcleo de B junto con las distribuciones de posibilidad (difusos) que dan compatibilidad 1 con B.

Nota:

Este resultado, si bien no reproduce exactamente el clásico *modus ponens* (es más general), no plantea graves inconvenientes dado que no parece muy razonable que sobre un mismo dominio se definan etiquetas solapadas a nivel 1, ya que éstas resultarían semánticamente ambiguas. Teniendo en cuenta estas consideraciones, añadiremos a la teoría la siguiente restricción de integridad:

Si $\mathcal{L}_A = \{A_1, A_2, \ldots, A_k\}$ es el conjunto de posibles etiquetas sobre el dominio U de un atributo difuso A, entonces debe cumplirse que

ET1.- $\forall\ (A_i, A_j),\ i \neq j,\ comp(A_i, A_j) < 1$

Una vez definida esta regla de integridad, no se permite que haya dos etiquetas definidas sobre el mismo dominio tales que se solapen en grado 1. El resultado, bajo esta condición, sería el conjunto formado por el propio B junto con los elementos de su núcleo, lo que reproduciría el modus ponens.

4.2.3 Proposiciones Condicionales Cualificadas Probabilísticamente

Estas proposiciones o reglas están basadas en hechos, esto es, se han obtenido de la experiencia, y son de la forma:

$$(Si\ X\ es\ A\ entonces\ Y\ es\ B)\ es\ \beta$$

donde β se entiende como la probabilidad de que se dé una propiedad B cuando se da A.

En esta situación, el esquema que describe inicialmente nuestro problema es el siguiente:

$$\frac{\begin{array}{l} Si\ (X\ es\ A\ entonces\ Y\ es\ B)\ es\ \beta \\ \quad X\ es\ A \end{array}}{(Y\ es\ B)\ es\ p}$$

donde p es la probabilidad de que se de $Y\ es\ B$, o lo que es igual, la probabilidad de que Y cumpla completamente la propiedad B.

En este caso, teniendo en cuenta que la probabilidad de partida de *"X es A"* es 1 (supuesto que es un hecho de la BD), se verifica que $p = \beta$

Si la premisa de la regla se ha obtenido por un proceso de propagación a través de reglas de tipo probabilístico, es posible que lleve a su vez asociada un valor de

probabilidad. En este caso, haciendo uso del teorema de Bayes para probabilidades condicionadas, obtendríamos que la probablidad final asociada al hecho Y *es* B se calcula como el producto de la probabilidad asociada a la premisa y la probabilidad asociada a la regla, bajo la hipótesis de independencia*.

Sin embargo, si en lugar de X *es* A tenemos en el antecendente X *es* A^*, obtendríamos para la premisa un valor de compatibilidad de naturaleza posibilística. A nuestro juicio, nos encontramos en una situación en la que no se ha descrito ninguna forma sensata de combinar ambas medidas (probabilidad y posibilidad). En el supuesto, que no es el caso, de que ambas medidas estuvieran asociadas al mismo hecho, ligar posibilidad y probabilidad supondría utilizar un marco más general en el que ambas tuvieran cabida (ver [38]), perdiéndose entonces el significado aportado por cada una de ellas. Pero nuestro problema es ligeramente distinto, ya que posibilidad y probabilidad se encuentran totalmente desligadas, esto es, cualifican distintos hechos*.

Teniendo en cuenta este valor de compatibilidad en el antecedente, obtenemos un valor de compatibilidad en el consecuente, o lo que es igual, un conjunto de posibles valores para B^*. Con esta interpretación, lo que obtenemos realmente tras la aplicación de una regla probabilística es un valor de probabilidad asociado a un conjunto.

Más formalmente, si del hecho "X *es* A^*" y la regla "*Si* X *es* A *entonces* Y *es* B" se obtiene el conjunto $\{B_1^*, B_2^*, \ldots, B_n^*\}$, cuando la regla lleva asociada una probabilidad p, la probabilidad resultante ha de asociarse a este conjunto, o lo que es igual, al resultado B_α (donde $\alpha = comp(A, A^*)$.

Así pues, la única afirmación que puede realizarse acerca de Y *es* B cuando se da X *es* A en un grado α, es que *"Y es compatible con B en grado* α *en el* $\beta * 100\%$ *de los casos"*, o equivalentemente, que la probabilidad asociada al conjunto $\mathcal{B}_\alpha$ es β.

4.2.4 Proposiciones Doblemente Cualificadas

Serían proposiciones cuya forma general es:

*Caso de no verificarse dicha independencia se debe suministrar información adicional al sistema

*Mientras que la posibilidad afecta directamente a las distribuciones premisa y conclusión, la probabilidad es independiente de la forma que adopten dichas distribuciones

$$((Si\ X\ es\ A\ entonces\ Y\ es\ B) es\ \alpha)\ es\ p$$

donde A y B pueden ser difusos.

En esta situación, el *esquema de inferencia* quedaría:

$$\frac{\begin{array}{l} Si\ ((X\ es\ A\ entonces\ Y\ es\ B)\ es\ \beta)\ es\ p \\ \quad (X\ es\ A)\ es\ \alpha \end{array}}{(Y\ es\ B\ es\ \nu)\ es\ p}$$

donde ν se construye como en los casos anteriores.

Obsérvese que, en este caso, no tiene ningún sentido la aplicación de un criterio de consistencia entre la probabilidad y la posibilidad, ya que ambas medidas se aplican a distintos conceptos (hechos); mientras que ν está asociada directamente al difuso B, p está asociada al conjunto descrito por la proposición $(Y\ es\ B\ es\ \nu)$, no a $Y\ es\ B$.

4.2.5 Definición Intensiva de Atributos

Gracias a la posibilidad de definir y utilizar *reglas definitorias*, se puede solicitar información a una BDLD acerca de una determinada propiedad, a verificar por los *objetos* en ella almacenada. Sin embargo, las posibilidades que ofrecen las reglas definitorias van más allá de las consultas, ya que, como ahora veremos, son de gran utilidad para ampliar de forma *virtual* una determinada relación de una BDLD.

Supóngase que con la información de que se dispone se puede definir (de forma más o menos precisa) toda una gama de etiquetas referentes a un mismo dominio o atri buto, y todas ellas de manera intensiva por medio de reglas definitorias. Por ejemplo, podemos plantearnos la definición de los predicados `alumno-malo`, `alumno-regular`, `alumno-mediocre`, `alumno-bueno`, `alumno-excelente` haciendo uso de información acerca de la edad, la calificación media, la participación en actividades docentes o investigadoras y la actitud mostrada en clase, de cada uno de los alumnos. En esta situación, podrían utilizarse cinco reglas definitorias, cada una de las cuales especificará las exigencias correspondientes con respecto a cada uno de los atributos que se

van a utilizar en las definiciones. Nótese que es posible que alguno de los atributos almacenados no se utilice en todas las definiciones.

En general, sea $\mathcal{A} = \{A_1, \ldots, A_l\}$ el conjunto de todos los atributos presentes en la BD y sea $\mathcal{L} = \{L_1, L_2, \ldots, L_n\}$ el conjunto de identificadores posibles para los elementos de un dominio adicional que se desea especificar. En esta situación, podemos dar la siguiente definición:

Definición 4.1 .

Diremos que A_V es un **atributo virtual difuso** *para una relación R si, y sólo si, se cumple que:*

(i) $\forall L_i \in \mathcal{L}, \exists \{A_i, \ldots, A_j\} \subseteq \mathcal{A}$ *tales que permiten definirlo (desde el punto de vista semántico).*

(ii) Cada uno de los elementos de $\mathcal{L}$ constituye la conclusión de, al menos, una regla definitoria que lleva asociada un umbral mínimo de posibilidad, previamente establecido, α. Con esta restricción aseguraremos una calidad *mínima y homogénea para todos valores del dominio y garantizamos, en definitiva, que dichas definiciones son sensatas.*

(iii) Ante un conjunto de valores $\{a_i, \ldots, a_j\}$ correspondientes a los atributos de un determinado individuo *$\{A_i, \ldots, A_j\}$, aplicaremos todas y cada una de las reglas de definición del dominio para el atributo virtual que se desee considerar, calculándose el valor actual de atributo virtual v_{A_V} como sigue:*

$$Sea\ L^{>0.5} = \{L_i \in \mathcal{L} \mid el\ valor\ \nu\ calculado\ verifica\ \nu \geq 0.5\}.$$

$$Entonces,\ v_{A_V} = L_1^{>0.5}/\nu_1 + \ldots + L_s^{>0.5}/\nu_s$$

Es decir, el valor que tomará el atributo virtual para dicho individuo será la distribución de posibilidad construida a partir de todas las etiquetas (conclusiones) cuya posibilidad asociada supere un umbral razonable (en este caso se ha considerado 0.5).

◇ Ejemplo

Podría plantearse la construcción del dominio `Calific-Alumno` donde el conjunto de posibles valores es `alumno-malo`, `alumno-regular`, `alumno-mediocre`, `alumno-bueno`, `alumno-excelente`

Para cada uno de estos valores de dominio virtual, será necesario dar, al menos, una regla que, en base a los atributos almacenados, nos dé una descripción del mismo. Por ejemplo,

(alumno(Nombre,Edad) ∧
matriculado(Nombre,_ Asig,Nota,Actitud) ∧
comp(Actitud,"Positiva",Pos1) ∧
mayor_ig(Nota,"Notable",Pos2) ∧
min(Pos1,Pos2,Pos))
⟶ alumno-excelente(Nombre,Asig).

(alumno(Nombre,Edad) ∧
matriculado(Nombre,_ Asig,Nota,Actitud) ∧
(comp(Actitud,"Buena",Pos1) ∧
mayor_ig(Nota,"Notable",Pos2)) ∨
comp(Actitud,"Normal",Pos1) ∧
mayor_ig(Nota,"Sobresaliente",Pos2)))
⟶ alumno-bueno(Nombre,Asig).
.....

Si en la base de hechos tenemos el alumno A_1, aplicaremos a sus atributos todas las reglas del dominio `Calific-Alumno`. Supongamos que los resultados obtenidos son:

$$alumno - excelente\ con\ grado\ 0.7$$

$$alumno - bueno\ con\ grado\ 0.75$$

$$alumno - mediocre\ con\ grado\ 0.8$$

$$alumno - regular\ con\ grado\ 0.4$$

Si el umbral impuesto es de 0.6, por ejemplo, la distribución resultante como valor del atributo `Calific-Alumno` para A_1 será:

$$alumno - excelente/0.7 + alumno - bueno/0.75 + alumno - mediocre/0.8$$

4.3 Mecanismo de Propagación

En esta sección, tratamos el problema de la propagación, esto es, cómo obtener un valor de posibilidad para un hecho que se ha deducido a partir de un conjunto de reglas de alguno de los tipos anteriores o de varios tipos a la vez.

En primer lugar, analizaremos conjuntos de reglas que puedan resultar conflictivos. Posteriormente, veremos cómo puede llevarse a cabo la propagación de incertidumbre cuando todas las reglas que intervienen son del mismo tipo y cuando se mezclan reglas de los tres tipos mencionados, en el mismo proceso de inferencia.

◇ Estudio de Reglas Conflictivas

En caso de producirse conflicto con respecto a la activación de **reglas posibilísticas**, esto es, si aparecieran dos o más reglas con distinta precondición pero igual conclusión, pueden darse dos casos:

* *Reglas con la misma conclusión pero con distinto grado asociado a la misma*, como por ejemplo:

$$Si\ X\ es\ A\ entonces\ (Y\ es\ B)\ es\ \alpha$$

$$Si\ X\ es\ C\ entonces\ (Y\ es\ B)\ es\ \beta$$

  En este caso, se aplicarán ambas reglas y se tomará en cada caso, aquélla que dé el valor más alto de posibilidad al resultado a la vista de la información actual de la BD.

* *Reglas que dan exactamente la misma conclusión y con el mismo grado*, como por ejemplo:

$$Si\ X\ es\ A\ entonces\ (Y\ es\ B)\ es\ \alpha$$

$$Si\ Z\ es\ C\ entonces\ (Y\ es\ B)\ es\ \alpha$$

En este caso, ambas reglas se sustituirán por una tercera equivalente a ellas de la forma:

$$Si\ (X\ es\ A)\ o\ (Z\ es\ C)\ entonces\ (Y\ es\ B)\ es\ \alpha$$

tomándose el máximo valor de compatibilidad entre los de las proposiciones que intervienen en la disyunción del antecedente.

En el caso de **reglas probabilísticas** pueden utilizarse criterios como aplicar todas las reglas, siempre que se verifique la precondición al nivel exigido, y dejar que el usuario, en función de la criticidad de la regla, decida o asuma alguno de los resultados. En el caso de reglas doblemente cualificadas, criterio a seguir podría ser el siguiente:

Supongamos las reglas:

$$(Si\ X_1\ es\ A_1\ entonces\ (Y\ es\ B)\ es\ \alpha_1)\ es\ p_1$$

$$(Si\ X_2\ es\ A_2\ entonces\ (Y\ es\ B)\ es\ \alpha_2)\ es\ p_2$$

$$\ldots\ldots$$

$$(Si\ X_n\ es\ A_n\ entonces\ (Y\ es\ B)\ es\ \alpha_n)\ es\ p_n$$

y supongamos que los factores de compatibilidad calculados definitivamente para $Y\ es\ B$ por cada una de las reglas son β_1,β_2,..., β_n y los de probabilidad son P_1, P_2, ..., P_n, respectivamente. En esta situación, se procederá como sigue:

* Si existe un par de reglas R_i y R_j tales que

  $\beta_i > \beta_j$ y $\P_i > \P_j$, se despreciará la información aportada por R_j.

* Entre las restantes reglas, se producirá una situación de conflicto entre los valores de posibilidad y de probabilidad, ya que en unos casos la primera medida será mayor en relación con la de otra regla, pero la segunda será menor y vicerversa. En esta situación, se darán como respuesta todas las soluciones posibles, de manera que no se desprecie la información aportada por cada una de las reglas. El usuario siempre podrá determinar cuál de las soluciones calculadas es la que más le interesa dependiendo de la importancia que le conceda a cada una de las medidas o bien utilizar alguna función de agregación creciente en ambas medidas y optimizarla, de manera que se permita dar un criterio de ordenación en $\mathcal{R}^2$.

4.3.1 Propagación a Través de Reglas Cualificadas Posibilísticamente

Si todas las reglas son de este tipo, el problema es sencillo, ya que todas las medidas involucradas, al ser de la misma naturaleza, pueden combinarse entre sí por medio de una T-Norma. Supongamos que tenemos un hecho en la BD, que llamaremos A^*, y el siguiente conjunto de reglas:

$$(A \longrightarrow B_1^*)\ \alpha_1$$

$$(B_1 \longrightarrow B_2^*)\ \alpha_2$$

$$(B_2 \longrightarrow B_3^*)\ \alpha_3$$

$$\ldots\ldots$$

$$\ldots\ldots$$

$$(B_n \longrightarrow B_{n+1}^*)\ \alpha_{n+1}$$

donde $comp(A, A^*) = \beta$ y $comp(B_i, B_i^*) = \beta_i$. En este caso, podremos afirmar que B_{n+1}^* es compatible con la respuesta en grado $TN(TN(\ldots\ (TN(TN(\ldots\ (TN(TN(TN(TN(TN(\beta, \alpha_1), \beta_1), \alpha_2)\ , \ldots, \beta_i)\ , \alpha_{i+1})\ , \ldots, \beta_n), \alpha_{n+1})$

Si se desea, pueden utilizarse dos T-Normas diferentes: una para combinar los factores de compatibilidad de las precondiciones, y otra para combinar este resultado con los grados de compatibilidad de las conclusiones previamente obtenidas.

4.3.2 Propagación a Través de Reglas Cualificadas Probabilísticamente

En este caso, tendremos que calcular, por un lado, el grado de posibilidad de la conclusión a partir de los grados de certeza de las precondiciones de las reglas que hayan sido activadas durante el proceso y, por otro lado, la probabilidad final de este hecho, a partir de las probabilidades asociadas a dichas reglas. Por ejemplo, supongamos que tenemos un hecho en la BD, que llamaremos A^*, y el siguiente conjunto de reglas:

$$(A \longrightarrow B_1^*)\ p_1$$

$$(B_1 \longrightarrow B_2^*)\ p_2$$

$$(B_2 \longrightarrow B_3^*)\ p_3$$

$$\ldots\ldots$$

$$\ldots\ldots$$

$$(B_n \longrightarrow B_{n+1}^*)\ p_{n+1}$$

donde $comp(A, A^*) = \beta$ y $comp(B_i, B_i^*) = \beta_i$. En este caso, podremos afirmar que B_{n+1}^* es compatible con la respuesta en grado $TN(\beta, \beta_1, \ldots, \beta_i, \ldots, \beta_n)$ y probabilidad $p_1 \times p_2 \times \ldots \times p_{n+1}$.

4.3.3 Propagación a Través de Cualquier Conjunto de Reglas

Si ambos tipos de reglas son considerados en el mismo proceso de inferencia, tendremos que distinguir, de alguna manera, cuándo se trata con una regla de tipo posibilístico y cuándo con una de tipo probabilístico. Usaremos la letra Π para denotar las posibilísticas y la letra P para las probabilísticas.

El proceso a seguir será como sigue:

* En cualquier paso del proceso, tendremos dos medidas: la *posibilidad* y la *probabilidad* acumuladas hasta el momento.

* Si en un momento dado tenemos el par (α, p), donde α es el grado de posibilidad y p el de probabilidad, la siguiente regla a aplicar es:

$$(B_i \longrightarrow B_{i+1}^*)\ \Pi = \alpha_{i+1}$$

y

$$comp(B_i, B_i^*) = \beta_i$$

el par (α, p) que se aplicará en el siguiente paso será:

$$\alpha = TN(TN(\alpha, \beta_i), \alpha_{i+1})$$

o bien

$$\alpha = TN_2(TN_1(\alpha, \beta_i), \alpha_{i+1})$$

y p queda como estaba.

* Si tenemos el par (α, p), la siguiente regla a aplicar es:

$$(B_i \longrightarrow B_{i+1}^*)\ P = \alpha_{i+1}$$

y

$$comp(B_i, B_i^*) = \beta_i$$

el par (α, p) que se aplicará en el siguiente paso será:

$$\alpha = TN_1(\alpha, \beta_i)$$

y

$$p = \alpha_{i+1} * p$$

* En el caso de la siguiente regla a aplicar sea doblemente cualificada, se llevarán a cabo las operaciones señaladas en los dos pasos anteriores simultáneamente.

En todos los casos presentados, si las precondiciones no son simples, esto es, contienen más de un predicado, calcularemos la conjunción por medio de una T-Norma (por ejemplo, el mínimo) y la disyunción por medio de una T-Conorma (por ejemplo, el máximo), para los valores de **posibilidad** de los predicados que aparezcan. Para

los valores de **probabilidad**, la conjunción se calculará por medio del producto* y la disyunción, como la diferencia entre la suma y el producto de las probabilidades asociadas a los predicados que aparezcan en la precondición.

En resumen, en esta sección hemos presentado los distintos tipos de reglas que podemos manejar y su significado. Así mismo, se han mostrado las características más relevantes del mecanismo de deducción a emplear y de sus posibilidades de aplicación en la extensión de la base de datos. Así pues, nos queda por mostrar cómo se puede representar todo el mecanismo de deducción junto con sus elementos (cualificación posibilística y probabilística, operadores, predicados difusos,....) dentro de nuestra definición lógica de base de datos difusa.

4.4 Definición Lógica de Reglas Difusas

En esta sección se plantea la modelización lógica de todos los elementos que intervienen en el mecanismo de deducción presentado en las secciones anteriores.

Cuando se pretende probar o deducir un hecho que no se encuentra de forma explícita en una BD a partir de la información almacenada en ella, será necesario activar una serie de reglas que nos conduzcan a conocer si se da o no dicho hecho para una serie de individuos de la BD. Cuando en el proceso de deducción, posiblemente en cadena, se activa cualquier tipo de regla en un momento dado, ésta puede modificar bien nuestra creencia sobre el hecho que se está intentando probar, bien la probabilidad de que este hecho se produzca, en función del tipo de regla que sea. En el caso más general, supongamos que ambos tipos de reglas son activados en el proceso de prueba de un determinado hecho. En este caso, el hecho resultante tendrá directamente asociado un valor mínimo de posibilidad y, este hecho junto con dicho valor, tendrán, a su vez, asociados un valor de probabilidad. Teniendo esto en cuenta, la forma general que tendrá cualquier *predicado definido de forma intensiva* será:

$$P(x_1, x_2, \ldots, x_n, \alpha, p)$$

*Bajo la hipótesis de independencia.

donde α es la posibilidad de P en función de la información de la que se dispone y, p es la probabilidad de que se dé $P(\alpha)$, calculada a partir de las probabilidades de cada uno de los predicados que intervienen en la definición de P.

A su vez, cada regla llevará asociados *obligatoriamente* dos predicados: **CD** (Compatibility Degree) para valorar su grado de validez, y **PB** (Probability) para valorar la probabilidad de que se produzca ese hecho. Gracias a estos dos predicados, podemos especificar cualquier tipo de regla, como sigue:

- **Reglas clásicas**: $CD(1)$ y $PB(1)$
- **Reglas definitorias**: $CD(k)$ con $k \leq 1$ y $PB(1)$
- **Reglas probabilísticas**: $CD(1)$ y $PB(k)$ con $k \leq 1$
- **Reglas doblemente cualificadas**: $CD(k_1)$ y $PB(k_2)$ con $k_1, k_2 \leq 1$

Según este esquema, para poder representar todos estos tipos de reglas, será necesario ampliar nuestro lenguaje, que incluirá dos nuevos nombres de predicado: CD y PB y añadir a la teoría un conjunto adicional de reglas de integridad para definir su comportamiento. Pasamos a verlo.

Definición 4.2 .

Llamaremos **Definición Intensiva Cualificada de Predicado Difuso** *a cualquier regla de la forma:*

$(CD(k_1) \wedge PB(k_2) \wedge A_1(x_{11}, x_{12}\ , \ldots, x_{1r}, \alpha_1, p_1) \wedge \ldots \wedge A_n(x_{n1}, x_{n2}, \ldots, x_{nm}, \alpha_n, p_n)$
$\wedge TN(k_1, \alpha_1, v_1) \wedge TN(v_1, \alpha_2, v_2) \wedge \ldots \wedge TN(v_{n-1}, \alpha_n, \alpha) \wedge *(k_2, p_1, t_1) \wedge \ldots \wedge *(t_{n-1}, p_n, p) \longrightarrow A(x_1, x_2, \ldots, x_n, \alpha, p))$

donde $A(\ldots)$ es siempre un predicado positivo (las reglas son cláusulas de Horn con cabeza). El valor de salida α tras aplicarse la regla, es una combinación de éste valor inicial con todos los grados de compatibilidad α_i en que se verifican cada uno de los antecedentes y es de tipo MD. El valor de salida p tras aplicarse la regla, se calcula como combinación de este valor inicial con todas las probabilidades p_i de cada uno de los antecedentes. Esquemáticamente, $\alpha = k_1 \circ \alpha_1 \circ \alpha_2 \circ \ldots \circ \alpha_n$ (donde $\circ$ equivale al predicado TN) y $p = k_2 * p_1 * p_2 * \ldots * p_n$

En principio, todos los hechos que se encuentran almacenados en la BD llevarán asociados $\alpha = 1$ y $p = 1$ por defecto, aunque esto no se especifique de manera explícita.

Definición 4.3 (Redefinición de Lenguaje Relacional Difuso)

Sea L un lenguaje $L = (A, W)$. Diremos que L es un **Lenguaje Relacional Difuso** *(LRD), si y sólo si:*

- *Incluye el tipo MD entre sus símbolos.*
- *Incluye los tipos CD y PB entre sus símbolos.*
- *Incluye, al menos, un símbolo de tipo difuso.*

Definición 4.4 .

Sea T una teoría relacional difusa, y sea T' tal que $T \subset T' \subseteq W$. Diremos que T' es una **Teoría Relacional Difusa con Proposiciones Doblemente Cualificadas (TRDPDC)** *si, y sólo si:*

(*i*) *T' incluye todas las fbf de T.*

(*ii*) *T' incluye, adicionalmente, las siguientes reglas de integridad para los predicados intensivos:*

* *La regla de integridad para las etiquetas lingüísticas* **ET1**, *que exige que dos etiquetas del mismo dominio no se solapen a nivel 1.*
* *Reglas para los predicados CD y PB*

  PI1.- $\forall x,\ CD(x) \longrightarrow \leq (0, x) \wedge \leq (x, 1)$.

  PI2.- $\forall x,\ PB(x) \longrightarrow \leq (0, x) \wedge \leq (x, 1)$.

* *Redefinición de los predicados TN y TCN:* Será necesario redefinir estos predicados, ya que ahora deberán poderse aplicar, no sólo a argumentos de tipo MD,

sino también a los de tipo CD.Para hacerlo, habrá que reformular nuevamente la regla **T1**.

T1'. $\forall\,(x,y,z),\ TN(x,y,z) \longrightarrow (MD(x)\vee CD(x))\wedge(MD(y)\vee CD(y))\wedge(MD(z)\vee CD(z)))$

El resto de las reglas formuladas para T-Normas y T-Conormas siguen siendo válidas.

* *Redefinición de los predicados $*$ y $+$:* Análogamente al caso anterior, habría que modificar las definiciones de los predicados $*$ (producto) y $+$(suma) que aparecen en la sección 3.3.2, para poder admitir argumentos de tipo PB, y añadir algunas reglas nuevas. La reformulación se indicará poniendo un apóstrofe junto a la clave de la regla a la que sustituye y la definición de nuevas reglas, análogas a algunas ya existentes, se indicará añadiendo a la clave una letra minúscula.

E5'.- $\forall(x,y,z),\ (+(x,y,z) \longrightarrow (MD(x)\wedge MD(y)\wedge MD(z))\vee(PB(x)\wedge PB(y)\wedge PB(z)))$

E9'.- $\forall x,\ (MD(x)\vee PB(x) \longrightarrow +(x,0,x))$

E10(b).- $\forall x,\ (PB(x) \longrightarrow \exists y,\ (PB(y)\wedge -(1,x,y)))$ *donde* $-$ *es el predicado asociado a la resta.*

E12(b).- $\forall(x,y),\ (PB(x)\wedge PB(y) \longrightarrow \exists z,\ (PB(z)\wedge *(x,y,z)))$

E15'.- $\forall x,\ (MD(x)\vee PB(x) \longrightarrow *(x,1,x))$

E16'.- $\forall x,\ (MD(x)\vee PB(x) \longrightarrow *(x,0,0))$

* *Reglas para los predicados intensivos:* Sea P_i un predicado intensivo. Entonces deberá verificar el siguiente conjunto de reglas:

PI3.- $\forall P_i,\ P_i(x_1,\ldots,x_n,y,z)\wedge \geq(n,1)$.

PI4.- $\forall(\alpha,y),\ P_i(x_1,\ldots,x_n,y,z) \longrightarrow \tau_1(x_1)\wedge\ldots\wedge\tau_n(x_n)\wedge MD(y)\wedge PB(z)$.

PI5.- $\forall(P_i,k_{1i}),\ (P_i(x_1,\ldots,x_n,y,z)\wedge CD(k_{1i}) \longrightarrow\leq(y,k_{1i}))$

PI6.- $\forall(P_i,k_{2i}),\ (P_i(x_1,\ldots,x_n,y,z)\wedge PB(k_{2i}) \longrightarrow\leq(p,k_{2i}))$

Estas dos últimas reglas se enuncian para establecer que los argumentos de los predicados CD y PB asociados a una regla, son los máximos valores permitidos acotar los valores de posibilidad y probabilidad del predicado resultado, respectivamente.

Definición 4.5 .

Llamaremos **BDRD Completa con Proposiciones Doblemente Cualificadas** *a un triple (R, T, RI) donde R es un lenguaje relacional difuso (según la nueva definición), T es una TRDPDC y RI incluye todas las reglas de integridad necesarias para la correcta manipulación de la BDD (incluyendo todas las descritas en esta sección).*

4.5 Conclusiones

En la primera parte de este capítulo, se han descrito desde un punto de vista general los mecanismos de propagación de incertidumbre que, a nuestro entender, se necesitan para definir conceptos intensivos en una BDRD lógica y para deducir información a partir de reglas dadas por un experto. Estas reglas pueden venir expresadas de manera imprecisa en términos probabilísticos o posibilísticos.

Para ello, se ha dado una interpretación de la posibilidad en términos de compatibilidades entre distribuciones de posibilidad, que, no sólo aporta una gran flexibilidad en las respuestas a las consultas sino que, adicionalmente, verifica buenas propiedades de cara a la deducción.

A partir de la definición intensiva de predicados, hemos visto cómo puede ampliarse el esquema de una relación construyendo nuevos dominios y definiendo nuevos atributos sobre ellos, cuyos valores se calculan automáticamente a partir de la información contenida en la BD.

En la segunda parte, se muestra cómo pueden expresarse tales *facilidades* desde el punto de vista de la lógica formal de predicados, construyendo una teoría especial como ampliación de la construida en el capítulo 5, y en la que los nuevos elementos a considerar así como su manipulación tienen cabida, completando así nuestra definición de BDRD lógica.

Este modelo lógico con capacidad de deducción, nos permite desarrollar sistemas de acoplamiento entre BDRD y un entorno de programación lógica de forma análoga a como se lleva a cabo en el caso clásico y que hemos descrito ampliamente en la sección 1.3.2. En nuestro caso, se introduce una complejidad adicional derivada del carácter impreciso de los datos y las reglas, complejidad que no nos impide, sin embargo, adoptar un entorno de programación lógica clásico como el Prolog. En el quinto y último capítulo de esta memoria, se muestra la arquitectura general de este sistema de acoplamiento que proponemos así como las características más relevantes de la implementación del modelo lógico.

Por último, señalar que cualquier tipo de cualificación de proposiciones no considerada aquí*, tiene perfecta cabida en nuestro modelo, sin más que añadir (o en su caso, rectificar) las restricciones correspondientes y las definiciones de los operadores que se utilicen. En concreto, la medida de necesidad no ha sido considerada por resultar demasiado restrictiva en nuestro contexto, llegando a ser, incluso, contraintuitiva (piense que la necesidad condicionada de A dado A es 0.5).

*Por ejemplo, en términos de *necesidad* o de *factor de certeza*

Capítulo 5

Características Generales de la Implementación: Ejemplo y Discusión

En este capítulo señalaremos las características más relevantes de la implementación de nuestro modelo lógico para BDD con posibilidades de deducción. Para ello, será necesario contar con un sistema que ofrezca suficiente generalidad y flexibilidad y que sea adecuado a nuestro contexto. En este sentido, parece bastante acertada la elección del lenguaje Prolog (lenguaje que se introdujo en el capítulo 1) y su entorno, ya que está establecido como una potente herramienta de investigación y desarrollo y, adicionalmente, nos permite tratar de una manera casi directa el problema que tenemos entre manos, pues se trata de un lenguaje de programación lógica. Aún así, será necesario construir todas las etiquetas, relaciones de semajanza, operadores y reglas difusas, así como adaptar el mecanismo de inferencia, en base a los elementos *"crisp"* que ofrece dicho sistema.

Terminaremos el capítulo con un ejemplo que nos permita ilustrar las características más interesantes del modelo junto con la correspondiente implementación en Delphia-Prolog*.

*Delphia-Prolog es una marca registrada de Sligos Agence Delphia (1992)

5.1 Esquema General de Acoplamiento entre BDR Difusas y Prolog

En el primer capítulo de esta memoria, se mencionó cómo se podían aunar las ventajas de las aproximaciones lógica y relacional de bases de datos clásicas, mediante una arquitectura adecuada que permitiera acoplar un sistema de programación lógica (como el Prolog) y un sistema de gestión de BD relacionales convencional. En este sentido, diversos autores han hecho su propuesta particular, de las cuales las más interesantes son las que se mencionan en la sección 1.3.2.

De manera análoga, resultaría altamente provechoso aunar las ventajas de las aproximaciones lógica y relacional de bases de datos difusas, salvo que hasta ahora no se contaba con ningún DBMS difuso comercial (si bien desde el punto de vista teórico, la producción científica ha sido muy abundante) ni tampoco de ningún sistema de programación lógica difusa de probada eficiencia.

Desde este punto de vista, haría falta, en primer lugar, el desarrollo de un SGBD relacional difuso eficiente y que verifique las características señaladas como deseables en la sección 2.2.2. y, en segundo lugar, un sistema de programación lógica como el que hemos propuesto en los capítulos anteriores, capaz de manejar tanto datos como operadores *difusos* y cuyo motor de inferencia sea capaz, a su vez, de admitir dichos datos junto con reglas imprecisas para deducir nueva información.

Así pues, una arquitectura idónea para llevar a cabo estos propósitos sería la que se muestra en la figura 5.1, donde se distinguen tres grandes módulos:

- *El SGBD difuso*: donde se llevarán a cabo todas las operaciones propias de un SGBD convencional, como inserción, borrado, actualización, indexación, etc...
- *El módulo de deducción*: constituido, básicamente, por un motor de inferencia adaptado a las nuevas necesidades. Toda la información a manejar deberá estar expresada en forma de hechos y reglas que, por lo general, serán imprecisos.
- *El módulo de control*: encargado de llevar a cabo las transferencias de control entre los dos módulos anteriores, de manera que se comuniquen de la forma más

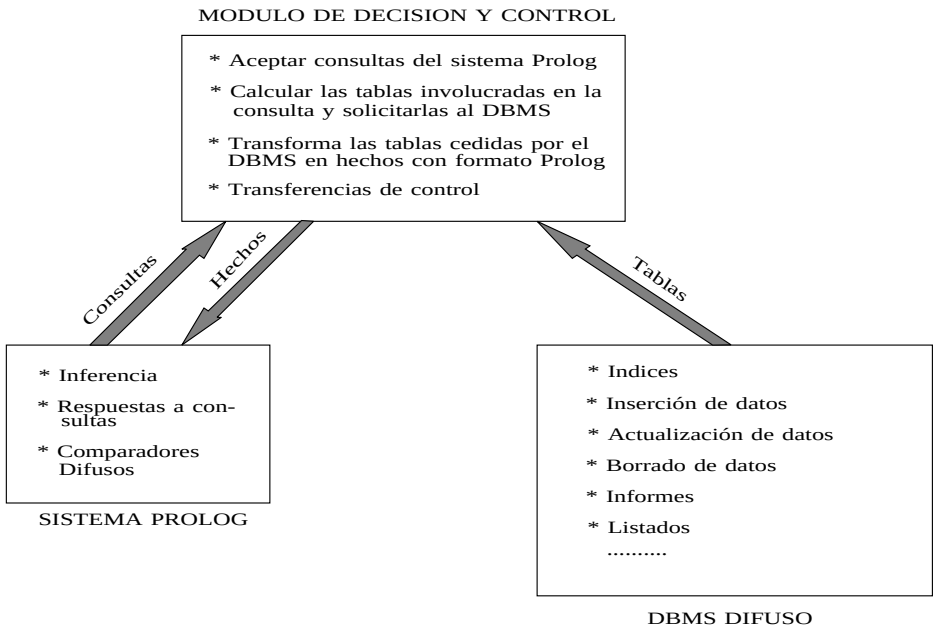


Figura 5.1. Arquitectura de un sistema difuso de acoplamiento DBRD-Prolog

eficiente posible. A su vez, tendrá que realizar las transformaciones de formato de los datos, de su estructura relacional a su estructura lógica, cada vez que se solicite la actuación del motor de inferencia para resolver una consulta.

En los capítulos tercero y cuarto, hemos llevado a cabo el desarrollo teórico del llamado módulo de deducción, que nos permitirá manipular tanto hechos como reglas en un entorno lógico que no difiere, en lo esencial, de los entornos lógicos utilizados para el caso clásico en los sistemas mencionados en el capítulo 1.

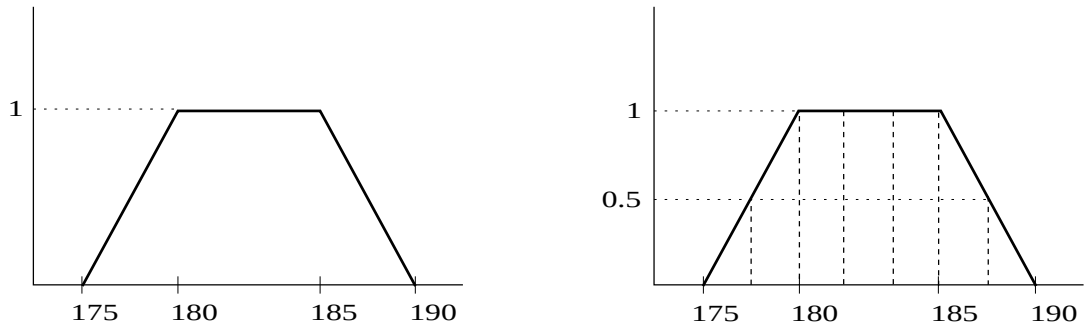
En este capítulo nos centramos en los aspectos más interesantes de la implementación de dicho módulo de deducción así como en algunos detalles del módulo de control. Para ello, previamente se analizan y se resuelven una serie de problemas computacionales que presenta el modelo teórico en cuanto a la representación de datos y operadores, pasando, una vez resueltos, a la implementación *efectiva* del modelo propuesto.

5.2 Características Generales de la Implementación del Modelo

Aunque nuestro modelo lógico teórico tiene muy buenas propiedades, también tiene algunos inconvenientes de cara a su implementación, como son:

* La representación de los datos difusos definidos sobre dominio contínuo y con representación contínua (etiquetas) se lleva a cabo por extensión, esto es, especificando qué valores va tomando la función de pertenencia correspondiente para un conjunto de elementos del dominio subyacente. Este inconveniente no genera sólo un problema de espacio en memoria, sino que además nos obliga a discretizar el dominio subyacente, de manera que en algunos casos, si el muestreo que se realiza sobre el dominio no es lo suficientemente *fino*, podría producirse una pérdida de información.

  Por ejemplo, si para la etiqueta *alto* tuviéramos la distribución de posibilidad de la figura 5.2(a),



a) Representación contínua de la etiqueta "ALTO".

b) Discretización de la etiqueta "ALTO".

Figura 5.2.

el modelo teórico nos obligaría a discretizar la distribución, como aparece, por ejemplo, en la figura 5.2(b). En esta situación, si apareciese en la BD o en alguna

consulta un individuo cuya altura es 176, ¿cuál sería la respuesta?. Obviamente nos veríamos obligados a llevar a cabo algún tipo de aproximación del valor que aparece en la consulta o desarrollar algún mecanismo específico para estos casos.

* De la misma manera que en el caso anterior, el modelo propuesto nos obliga a definir todos los comparadores -difusos o no- de forma extensiva.

* Existe mucha información redundante debido a la simetría de muchas de las relaciones (predicados) que aparecen.

Como puede verse, el problema principal con el que nos encontramos se debe a la representación tanto de datos como de operadores, lo que nos obliga a buscar representaciones alternativas equivalentes que no presenten dichos inconvenientes.

5.2.1 Representación Lógica Unificada de Datos Difusos

En esta sección vamos a establecer qué tipos de datos difusos se van a considerar y qué predicados será necesario incluir para su representación.

◇ **Datos Difusos sobre Dominio Contínuo**

• Etiqueta Lingüística

Este tipo de dato se representará por medio de una función trapezoidal junto con un nombre. Tal función queda perfectamente definida por cuatro parámetros de la forma (a, b, c, d), como se ve en la figura 5.3(a).

• Tipo Aproximado

Aquí se consideran aquellos datos de los que no se conoce el valor exacto, pero se tiene una idea o aproximación del mismo. Estos datos pueden representarse por medio de una función triangular, de manera que su representación puede ser idéntica a la de la etiqueta lingüística haciendo $b = c$, como se ve en la figura 5.3(b). A este tipo de dato se le denomina también *constante difusa* en [117].

• Tipo Intervalar

Este tipo representa aquellos datos de los que no se conoce el valor exacto pero sí los límites inferior y superior entre los que se encuentra. Igual que en los casos anteriores, la representación puede llevarse a cabo por medio de los parámetros (a, b, c, d) haciendo $a = b$ y $c = d$, como se ve en la figura 5.3(c).

Como puede verse, es suficiente con dar cuatro parámetros y un nombre para tener perfectamente definido cualquier dato de uno de los tipos anteriores. Esta representación facilitará enormemente tanto su almacenamiento como su manejo. Para llevar a cabo la implementación en Prolog de los mismos, introduciremos un nuevo predicado, **LABEL**, que asociará una etiqueta de un determinado dominio con sus correspondientes parámetros. Por ejemplo:

$$LABEL(Edad, "Media", 30, 35, 40, 45)$$

$$LABEL(Edad, "30-35", 30, 30, 35, 35)$$

$$LABEL(Edad, "Aprox.\ 47", 45, 47, 47, 49)$$

donde "$Media$" es una etiqueta lingüística, "$30-35$" es un intervalo y "$Aprox.\ 47$" es un tipo aproximado.

Todos los tipos de datos mencionados hasta ahora, se suponen normalizados.

• Distribución de Posibilidad

Este tipo de dato se introduce para representar conocimiento impreciso sobre dominio contínuo, pero donde el dato, en sí mismo, no tiene una representación contínua. Esto es, el dato puede tomar diferentes valores del dominio subyancente con distintos valores de posibilidad, pero no tiene por qué barrer todo el dominio. Los datos de este tipo se representar por distribuciones de la forma :

$$v_1/c_1 + v_2/c_2 + \ldots + v_n/c_n$$

donde v_i/c_i significa que el dato en cuestión toma el valor v_i con posibilidad c_i.

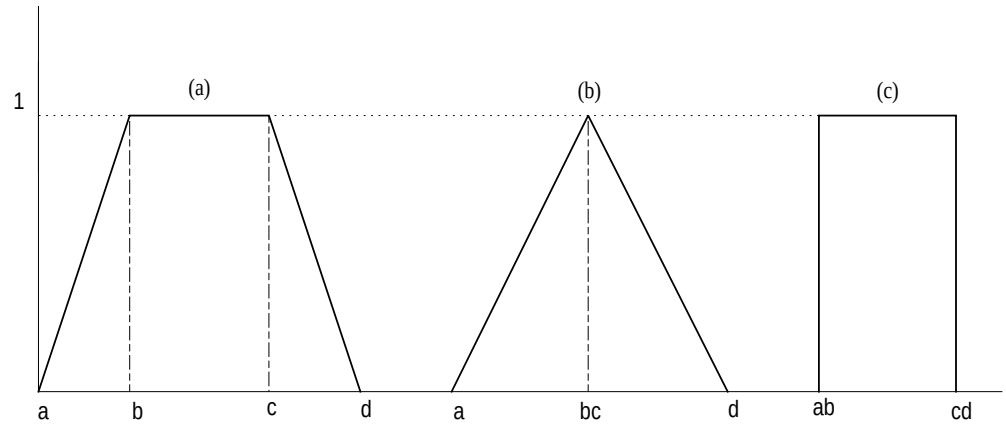


Figura 5.3. Representación de tipos de datos difusos contínuos

Con este tipo de datos, sin embargo, se nos plantea el problema* de que una distribución de posibilidad no tiene un número fijo de componentes. Para resolverlo, obligaremos a que toda distribución de posibilidad lleve asociado un nombre, con o sin significado, de la siguiente forma:

Si tenemos una distribución de posibilidad de la forma

$$n = v_1/c_1 + v_2/c_2 + \ldots + v_n/c_n$$

como valor de un atributo A_i perteneciente a una relación de esquema $(A_1, A_2, \ldots, A_n)$ tendremos, por un lado, el predicado correspondiente a la tupla a la que dicho valor pertenece - $TABLA(a_1, a_2, \ldots, n, \ldots, a_n)$- y, por otro lado, la composición de la distribución n. Para llevar a cabo esto último será necesario introducir un nuevo predicado que llamaremos **POS** y que nos relacionará n con cada uno de sus componentes de la siguiente manera:

$$POS(\tau, n, v_1, c_1)$$

$$POS(\tau, n, v_2, c_2)$$

*Computacionalmente hablando

.....

$$POS(\tau, n, v_n, c_n)$$

donde τ es el atributo al que corresponde la distribución especificada, y se añadirán tantas instancias del predicado POS como valores aparezcan en la distribución. Por ejemplo, si tenemos una relación ALUMNO(Nombre,Edad,Curso,Comportamiento) y un valor para la edad correspodiente a la distribución $A1 = 35/1 + 36/1 + 37/0.9$ como representación de la información *"La edad de X es 35 con posibilidad 1, 36 con posibilidad 1 ó 37 con posibilidad 0.9"*, la representación correspondiente en términos de Prolog será:

$$ALUMNO(X, A1, 5, Regular)$$
$$POS(Edad, A1, 35, 1)$$
$$POS(Edad, A1, 36, 1)$$
$$POS(Edad, A1, 37, 0.9)$$

◇ **Datos Difusos sobre Dominio Discreto**

• **Distribución de Posibilidad**

Este tipo de dato se define para poder representar datos imprecisos sobre dominios no contínuos. El mecanismo de representación y el significado es exactamente el mismo que el que tienen las distribuciones para el caso de domínio contínuo. Así pues, se hará uso del predicado POS.

Es importante señalar, que todos estos predicados son *atributo-dependientes*, ya que dos valores de atributo pertenecientes a distintos dominios pueden tener el mismo nombre pero asociado a distintos parámetros.

5.2.2 Algoritmo para la Representación Lógica de Datos

Como se vio en el esquema general de nuestro sistema (cap. 2), partimos de un sistema de gestión de bases de datos relacionales difusas en el que la información está representada por medio de tablas. La primera tarea a realizar por el módulo de control de

ambos sistemas es expresar dicha información de forma que pueda ser procesada por un sistema lógico, esto es, convertirla en expresiones lógicas equivalentes en el formato Prolog.

Dada una relación T de una BD, la información en ella contenida será representada de la siguiente forma:

1. Para cada tupla de la relación, se incluirá el predicado $T(a_1, a_2, ..., a_n)$ a la base de hechos de Prolog.

2. Para cada atributo de la tabla y para cada uno de los valores que éste tome en las tuplas, se añadirá el predicado $A_i(a_j),\ i = 1, ..., n\ j = 1, ..., m$ donde A_i es el i-ésimo atributo de la tabla y a_j son los valores que, para dicho atributo, existen en la tabla. Así pues, en este paso se establecen los dominios actuales para los atributos de una tabla concreta de una BD.

3. Repetir los pasos 1) y 2) para cada una de las tablas de la BD.

   Como habrá información redundante en una BD (por ejemplo, los valores de llave externa deberán coincidir con los de alguna llave primaria), se deberá incluir en el algoritmo el siguiente paso:

4. Eliminar todos los predicados redundantes originados en el paso 2). Con estos cuatro pasos, tenemos ya representada toda la información relevante de la BD. Sin embargo, será necesario también incluir todo el metaconocimiento referente a los valores de atributos difusos. Por ello, en último lugar habrá que:

5. Añadir todas las definiciones de los datos difusos que aparecen en la BD original por medio de los predicados LABEL y POS mencionados previamente.

   Una vez que la ejecución del algoritmo ha finalizado, tenemos completa la BD del entorno de Prolog.

Veamos cómo quedaría la tabla 3.1 tras la aplicación del anterior algoritmo.

– *Paso 1*:

$ALUMNO(Jose, Muy\ Joven, 1, Regular)$
$ALUMNO(Maria, 14, 1, Bueno)$
$ALUMNO(Antonio, Joven, 2, Muy\ Bueno)$
$ALUMNO(Carlos, 16, 3, Malo)$
$ALUMNO(Susana, Media, 5, Ejemplar)$

– *Paso 2*:

$NOMBRE(Jose)$	$EDAD(Muy\ Joven)$	$CURSO(1)$
$NOMBRE(Maria)$	$EDAD(14)$	$CURSO(1)$
$NOMBRE(Antonio)$	$EDAD(Joven)$	$CURSO(2)$
$NOMBRE(Carlos)$	$EDAD(16)$	$CURSO(3)$
$NOMBRE(Susana)$	$EDAD(Media)$	$CURSO(5)$

$COMP(Regular)$
$COMP(Bueno)$
$COMP(Muy\ Bueno)$
$COMP(Malo)$
$COMP(Ejemplar)$

– *Paso 3*:

Eliminar todos las instancias de predicado redundantes.

– *Paso 4*:

$LABEL(Edad, "MuyJoven", 11, 15, 18, 22)$

$LABEL(Edad, "Joven", 20, 24, 28, 32)$

$LABEL(Edad, "Media", 30, 32, 36, 40)$

y, adicionalmente, las etiquetas que posiblemente se vayan a utilizar en las consultas, como:

$LABEL(Edad, "Madura", 38, 42, 46, 50)$

$LABEL(Edad, "Mayor", 48, 52, 56, 60)$ .....

Nótese que hemos omitido algunos puntos del modelo teórico (axioma de nombre único, relación de orden para el tipo MD,...) ya que se verifican por defecto en el entorno de Prolog.

5.2.3 Operadores Relacionales Difusos

Una de las facilidades más importantes que debe incluir un sistema que maneje información difusa, es la posibilidad de realizar consultas, ya sean precisas o difusas, a la BD para obtener información relevante. En este sentido, lo primero que se debe resolver es el problema de las comparaciones, esto es, definir los operadores relacionales que nos permitan establecer relaciones de orden entre valores de atributo precisos y difusos y calcular hasta qué punto dicha relación se verifica.

En nuestro caso, hemos definido e implementado los siguientes operadores relacionales difusos:

- **COMP(X,Y,Z)**: Es equivalente al comparador de igualdad pero extendido al caso difuso. Los dos primeros argumentos son los valores a comparar y el tercero, el grado en el que se verifica la compatibilidad entre ambos argumentos. Al comparar dos atributos, pueden darse tres casos:

  1. *X e Y son "crisp"*: El comportamiento del operador es el de la igualdad clásica.
  2. *X ó Y (o ambos) es una etiqueta*: Para ello, ambos valores habrán de pertenecer a un dominio contínuo y uno de ellos, al menos, ser de tipo *"label"*. En este caso, el grado de compatibilidad entre ambos es calculado utilizando la definición de la etiqueta que se encuentra almacenada en la BD como predicado LABEL. Para calcularlo, puede optarse por calcular la *posibilidad* de X conocido Y, por medio de la expresión:

     $$Sup_u\{Min\{X(u), Y(u)\}\}$$

     como se muestra en la figura 5.4.
  3. *X e Y tienen dominio discreto*: En este caso, el grado de compatibilidad entre ambos valores no puede ser calculado, sino que deberá ser consultado en alguna tabla donde se especifique el grado de similaridad existente entre cada dos valores del dominio. Esta tabla será implementada en nuestro entorno por medio de un nuevo predicado, **SIMIL**, que posee tres argumentos: los dos

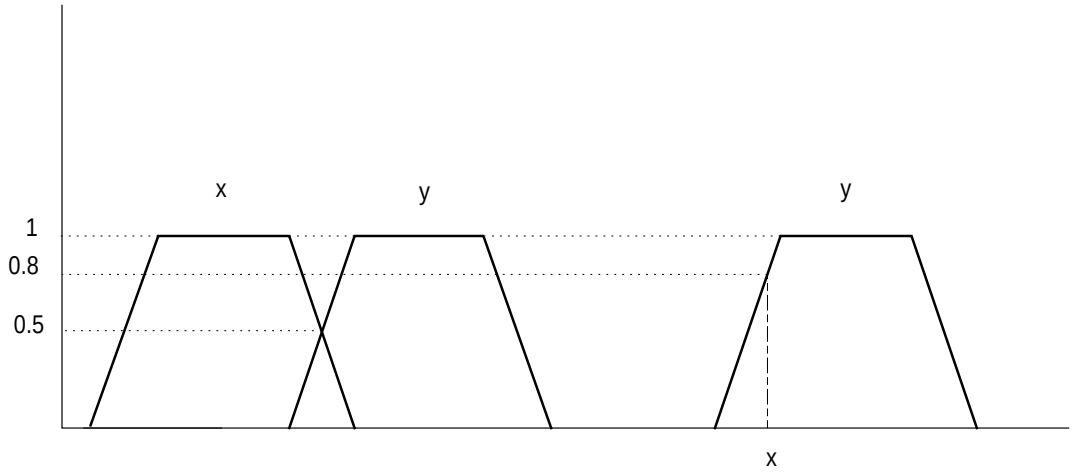


Figura 5.4. Comportamiento del operador de compatibilidad sobre etiquetas

valores a comparar y el grado de similaridad entre ellos. Así pues, el resultado del predicado $COMP$ se obtendrá simplemente consultado la instancia adecuada de dicho predicado.

Para nuestro ejemplo, podrían añadirse, por ejemplo, los predicados de la tabla 5.1:

En este último caso, será necesario especificar de alguna forma la propiedad simétrica del predicado $SIMIL$. En el modelo teórico, esto se lleva a cabo duplicando el número de instancias de dicho predicado, pero cambiando el orden del

$SIMIL(Comportamiento, Malo, Regular, 0.8)$
$SIMIL(Comportamiento, Malo, Bueno, 0.3)$
$SIMIL(Comportamiento, Regular, Bueno, 0.7)$
$SIMIL(Comportamiento, Regular, Muy\ Bueno, 0.2)$
$SIMIL(Comportamiento, Bueno, Muy\ Bueno, 0.9)$
$SIMIL(Comportamiento, Bueno, Ejemplar, 0.4)$
$SIMIL(Comportamiento, Muy\ Bueno, Ejemplar, 0.9)$

Tabla 5.1. Instancias del predicado SIMIL

segundo y tercer argumentos. En nuestro caso, bastará con añadir al sistema la siguiente regla:

$$COMP(X,Y,G) :- SIMIL(X,Y,G) \; or \; SIMIL(Y,X,G)$$

que significa que cuando se comparan dos valores definidos sobre dominio discreto, el orden en el que aparecen los argumentos es irrelevante.

Por otro lado, la propiedad reflexiva, que obligaría a relacionar cada valor consigo mismo en grado 1, quedará expresada por medio de la regla:

$$SIMIL(X,X,G) :- G = 1$$

que significa que, si los argumentos son idénticos, entoces el grado G es directamente igual a 1.

- **MAYOR_IG(X,Y,Z)**: Es la extensión del operador *mayor o igual* al caso difuso. Calcula el grado en el que se verifica que X es mayor o igual a Y y lo almacena en Z. Para argumentos "crisp" el comportamiento es el clásico y para el caso en el que alguno de ellos o ambos sea difuso, se calculará -de forma temporal- la etiqueta *mayor o igual que* Y y se calculará su compatibilidad con X, como aparece en la figura 5.5.

- **MENOR_IG(X,Y,Z)**: Este caso es análogo al anterior, ya que se construye la etiqueta correspondiente a *menor o igual que* Y y se compara con X. Ver fig. 5.6.

  Los operadores *mayor* (**MAYOR(X,Y,Z)**) y *menor* (**MENOR(X,Y,Z)** se implementan utilizando los dos operadores anteriores como sus correspondientes complementarios de la siguiente forma:

$$MAYOR(X,Y,Z) :- MENOR\_IG(X,Y,Z1) \; and \; Z = 1 - Z1.$$

como complemento del operador $MENOR\_IG$, y

$$MENOR(X,Y,Z) :- MAYOR\_IG(X,Y,Z1) \; and \; Z = 1 - Z1.$$

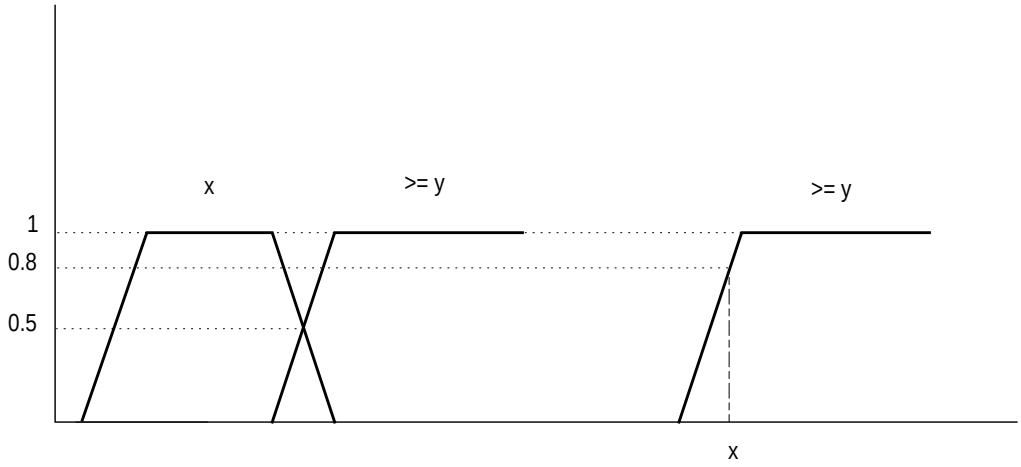


Figura 5.5. Comportamiento del operador *"mayor o igual"* difuso.

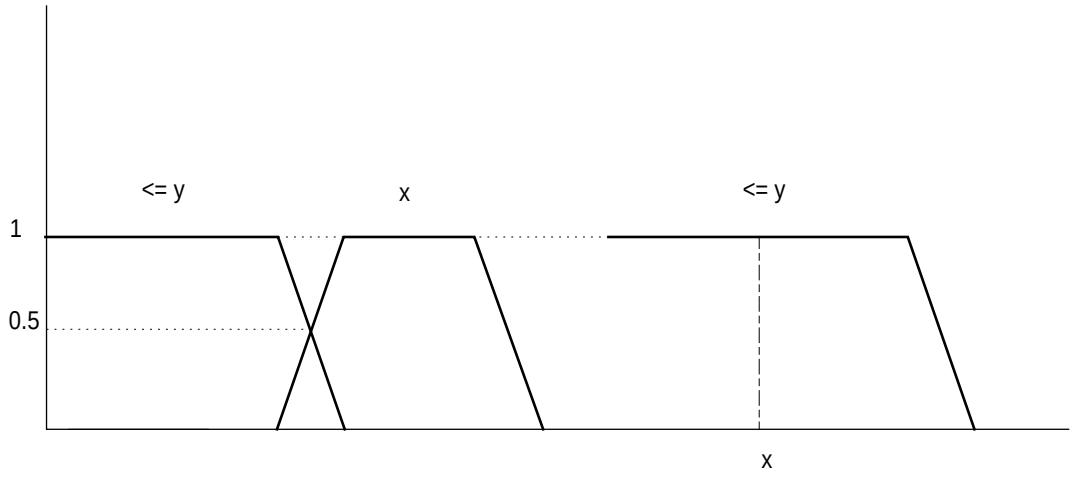


Figura 5.6. Comportamiento del operador *"menor o igual"* difuso

como complemento del operador $MAYOR\_IG$.

En el primer caso, X es mayor que Y con grado Z si X es menor o igual a Y con grado $Z1$ y Z vale $1 - Z1$. El segundo caso es análogo pero utilizando los operadores $MENOR$ y $MAYOR\_IG$.

5.2.4 Especificación de Consultas

Una vez adoptadas tanto la representación de los datos como la de los operadores previamente presentadas, estamos en disposición de consultar la BD, teniendo en cuenta los siguientes puntos:

- Para cada atributo que interviene en la consulta, puede especificarse un grado de cumplimiento individual*.
- Tras una consulta, se seleccionarán aquellas tuplas cuyo grado de cumplimiento para cada uno de los atributos involucrados sea igual o mayor a los grados especificados en la consulta.
- Cuando no se especifique grado de cumplimientos para alguno de los atributos involucrados en una consulta, se supondrá que se exige mayor que cero.

Veamos algunos ejemplos:

* *"Seleccionar aquellos alumnos cuya edad es menor o igual a joven y cuyo comportamiento es más o menos bueno"*

ALUMNO(Nombre,Edad,_,Comport) and MENOR_IG(Edad,"JOVEN",G1) and G1>0 and COMP(Comport,"BUENO",G2) and G2>0.

Las tuplas seleccionadas son:

*Para los grados de cumplimiento de las condiciones exigidas, usaremos la notación G_i

Nombre	**Edad**	**Curso**	**Comportam.**
Jose	*Muy Joven* (1)	1	*Regular* (0.7)
Maria	14 (1)	1	*Bueno* (1)
Antonio	*Joven* (1)	2	*Muy Bueno* (0.9)
Carlos	16 (1)	3	*Malo* (0.3)
Susana	*Media* (0.35)	5	*Ejemplar* (0.4)

* *"Seleccionar aquellos alumnos cuya edad es mayor a 25 años con grado 0.8 y cuyo comportamiento es regular con grado 0.7"*

ALUMNO(Nombre,Edad,_Comport) and MAYOR(Edad,"25",G1) and G1>0.8 and COMP(Comp,"REGULAR",G2) and G2>0.7.

Las tuplas seleccionadas son:

Nombre	**Edad**	**Curso**	**Comportam.**
Antonio	*Joven* (1)	2	*Muy Bueno* (0.2)

En ambos casos, el valor que aparece entre paréntesis junto al valor de atributo correspondiente, es el grado en el que se verifica la propiedad exigida en la consulta.

Si se quisiera hacer consultas imprecisas sobre el atributo CURSO (que es *crisp*), habría que incluir las relaciones de semejanza correspondientes a las etiquetas utilizadas en las consultas y los valores permitidos en la relación para el atributo CURSO, esto es, los predicados de la tabla 5.2. para el conjunto de etiquetas $\{Bajo, Medio, Alto\}$.

Como puede verse, es muy sencillo consultar una BD difusa de esta forma, ya que sólo es necesario conocer los nombres y parámetros de los operadores relacionales difusos definidos y cómo se construyen expresiones correctas en el cálculo relacional.

5.3 Características Generales de la Definición de Reglas Difusas

En las secciones anteriores, hemos explicado cómo se llevaría a cabo la representación de los datos existentes en una o varias relaciones y cómo se definen los operadores para

$$
\begin{array}{l}
SIMIL(Bajo, 1, 1) \\
SIMIL(Medio, 2, 0.6) \\
SIMIL(Medio, 3, 1) \\
\ldots\ldots \\
SIMIL(Alto, 5, 1) \\
SIMIL(Alto, 6, 1) \\
SIMIL(Bajo, Medio, 0.6) \\
SIMIL(Medio, Alto, 0.7)
\end{array}
$$

Tabla 5.2. Relación de similitud para el atributo CURSO

llevar a cabo consultas sobre dichos datos. Nos queda, pues, por comentar cómo llevar a cabo el proceso de deducción partiendo de una base de hechos y operadores como la que hemos mencionado, teniendo en cuenta los tipos de reglas y la forma de llevar a cabo las deducciones, tal y como se mencionan en el capítulo 4.

A la hora de llevar a cabo la implementación de las reglas, nos encontramos con que éstas han de ser, necesariamente, cláusulas de Horn con cabeza, esto es, un literal positivo como conclusión o cabeza de la regla y varios literales como precondición o cuerpo de la regla (Ver secc. 1.2.2). Esto evita que puedan producirse inconsistencias y no supone ninguna restricción, ya que para conocer el valor de certeza de la negación de un literal, se recurrirá al operador NEG (tomado como $1 - x$) que se define en 3.6.1. Nótese que es diferente preguntar por la *posibilidad* de que se dé $\neg\ P$ que por la *probabilidad* de que se dé $\neg\ P(\alpha)$, ya que en el primer caso habrá que aplicar el operador NEG al valor de posibilidad asociado a P y, en el segundo, al valor de probabilidad asociado a la proposición $P(\alpha)$. Esto es, si el predicado P tiene la forma:

$$P(x_1, x_2, \ldots, x_n, \alpha, p)$$

$$Poss(\neg\ P) = NEG(\alpha, n_1)\ mientras\ que\ Prob(\neg\ P(\alpha)) = NEG(p, n_2)$$

.

Adicionalmente, nosotros impondremos que cada regla difusa que se defina lleve, obligatoriamente, dos argumentos de salida (α y p) que se calcularán en el propio cuerpo de la regla a partir de los correspondientes valores de posibilidad y probabilidad de todas las reglas activadas en el proceso.

En la siguiente sección mostraremos algunos ejemplos de reglas difusas, definidas en base a un conjunto más o menos amplio de datos, de manera que puedan obtenerse resultados más realistas de su aplicación.

5.4 Ejemplo

Con el fin de ilustrar ampliamente tanto los mecanismos de representación de datos y operadores, como la definición de reglas y el proceso de deducción, presentamos a continuación la completa implementación del modelo, partiendo de los datos de los alumnos matriculados en la Diplomatura de Informática. Se consideran tres relaciones, que son:

- **ALUMNO**: En ella se especifican los datos personales del alumno. Por simplificar se han considerado los atributos **Nombre**, **Edad**, y **Provincia**. El atributo **Edad** es difuso y admite valores *"crisp"*, etiquetas y distribuciones de posibilidad. (Tabla 5.4).
- **ASIGNATURA**: En ella se especifican las características de cada asignatura. Se han considerado los atributos nombre de **Asignatura**, número de **Créditos** y **Curso** en el que se imparte. (Tabla 5.5).
- **MATRICULADO**: En ella se especifican las matrículas de los alumnos por asignaturas y las calificaciones obtenidas en cada una de ellas. Los atributos considerados son: **Nombre**, **Grupo**, **Asignatura**, **Nota** y **Actitud**, donde **Nota** es un atributo difuso, y **Actitud**, siendo *"crisp"*, lleva asociada una relación de semejanza. (Tabla 5.6).

Estas tres relaciones junto con los datos que contienen serán las que utilizaremos en todos los ejemplos que se describen a continuación y aparecen de forma explícita al

final del capítulo.

En cualquier caso, aunque es una visión muy simplificada de la gestión de los alumnos, esperamos que resulte lo suficientemente instructiva para poner de manifiesto las cualidades de nuestro modelo e implementación, que es nuestro objetivo.

A continuación, pasamos a ver los conjuntos de etiquetas que se han definido para manejar los atributos difusos mencionados.

En 5.7 a) aparecen las descripciones de las etiquetas permitidas en consultas para la edad. Las etiquetas no se han definido en base a la edad absoluta de los alumnos sino en base a la edad universitaria.

En 5.8 se describen dos etiquetas para la edad que aparecen en la relación ALUMNO, correspondientes al intervalo $[17 - 19]$ la primera y al valor *Aproximadamente 21* la segunda.

En 5.7 b) se muestran las etiquetas definidas para la calificación de los alumnos y en 5.3 la relación de semejanza entre los valores permitidos para la actitud.

	Muy Neg	**Negativa**	**Normal**	**Positiva**	**Muy Pos**
Muy Neg	1	0.7	0	0	0
Negativa	0.7	1	0.4	0	0
Normal	0	0.4	1	0.7	0.5
Positiva	0	0	0.7	1	0.8
Muy Pos	0	0	0.5	0.8	1

Tabla 5.3. Relación de Semajanza para el atributo Actitud

En base a todos estos datos, vamos a considerar la definición de los siguientes enunciados:

* *"Un alumno se considera bueno en una asignatura si tiene la edad adecuada al curso que corresponde a dicha asignatura, si muestra una actitud positiva y si la calificación es de notable como mínimo".*

* *"Un curso es posiblemente difícil* * *si tiene un número alto de asignaturas y con un número alto de créditos".*

  Obsérvese que la definición no es demasiado fina, ya que no se contempla la dificultad propia del temario, ni si tiene prácticas, ni los conocimientos previos aconsejables, etc... Esta falta de datos se pondrá de manifiesto en la asignación de un valor de incertidumbre *a priori* a la conclusión.

* *"Una asignatura es mala*† *con probabilidad* p *si pertenece a un curso difícil, si la proporción de alumnos buenos matriculados en ella es baja y si supera un número determinado de alumnos".*

  Al asignar una probabilidad a este hecho, damos por supuesto que se ha llevado a cabo algún tipo de estudio estadístico o que la naturaleza del valor asociado es frecuentista. al respecto.

  El cálculo de la proporción de alumnos buenos matriculados en una determinada asignatura, se calculará según la expresión:

*Esta cualificación posibilística deberá traducirse a un valor.

†Desde el punto de vista del que la imparte.

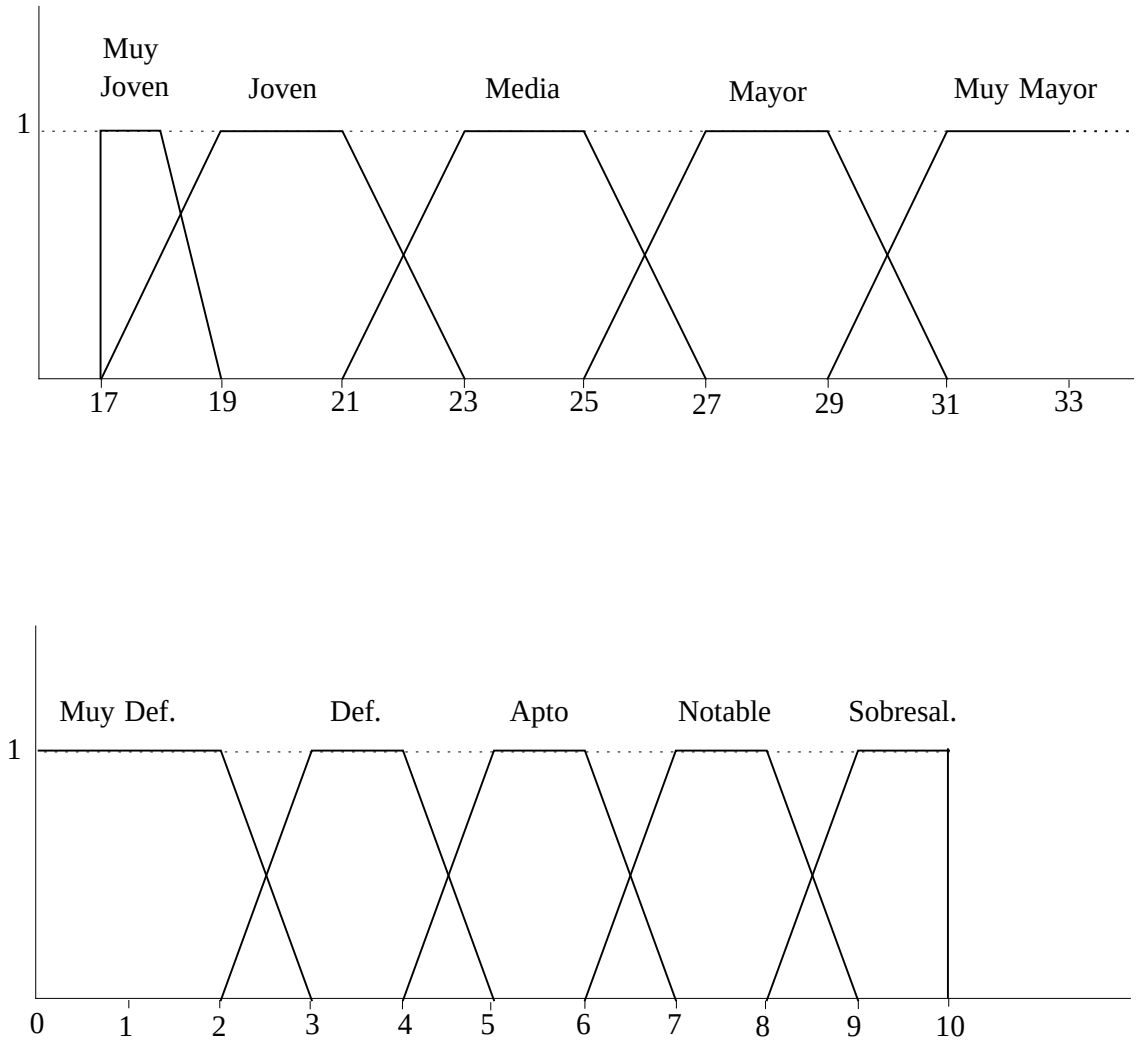


Figura 5.7. Etiquetas correspondientes a EDAD y NOTA

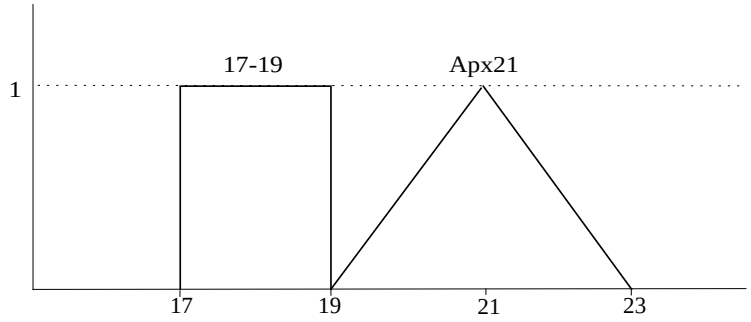


Figura 5.8. Etiquetas especiales correspondientes a la EDAD

$$\sum -Suma(B/A) = \frac{\sum_j (\mu_A(u_j) \wedge \mu_B(u_j))}{\sum_j \mu_A(u_j)} \ , \ j = 1, \ldots, m.$$

esto es, la $\sum -suma\ relativa$, donde A es el conjunto de los alumnos matriculados en la asignatura en cuestión, B el de alumnos buenos matriculados en dicha asignatura y $\mu_A(u_j)$ y $\mu_B(u_j)$ son los grados de pertenencia de u_j a los conjuntos A y B respectivamente.

Para poder llevar a cabo la definición de estos enunciados como reglas de Prolog, es necesario definir algunas etiquetas adicionales a las que aparecen almacenadas como datos en las relaciones.

Por ejemplo, hará falta conocer qué edades son las adecuadas a cada curso. Para ello, se han definido tres etiquetas sobre la edad: $Edad1$, que sería la que define la edad adecuada de los alumnos de primero, $Edad2$, que sería la adecuada para los de segundo y $Edad3$, para los de tercero. La definición de las mismas puede verse en la figura 5.9.

Con esta misma idea, para conocer si el número de créditos totales de un determinado curso es o no alto, se han definido las etiquetas $Bajo$, $Medio$ y $Alto$, tal y como aparecen en la figura 5.10 y también las etiquetas para poder determinar cuándo una

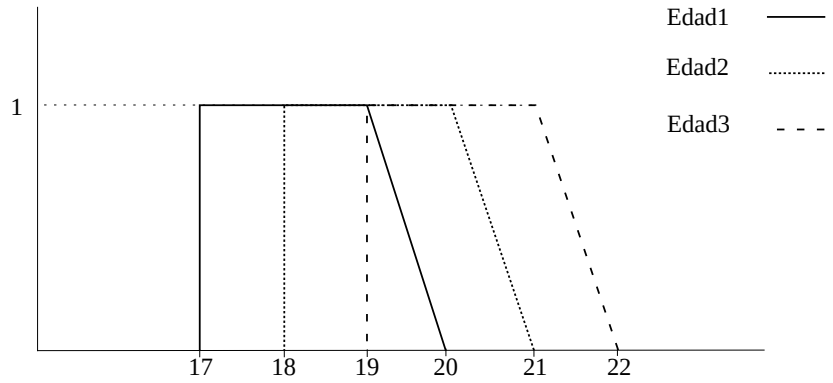


Figura 5.9. Etiquetas de las edades que corresponden a cada curso

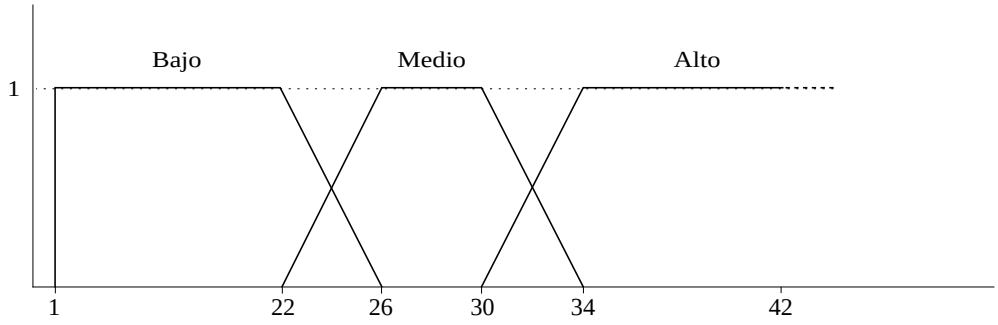


Figura 5.10. Etiquetas correspondientes al número de créditos

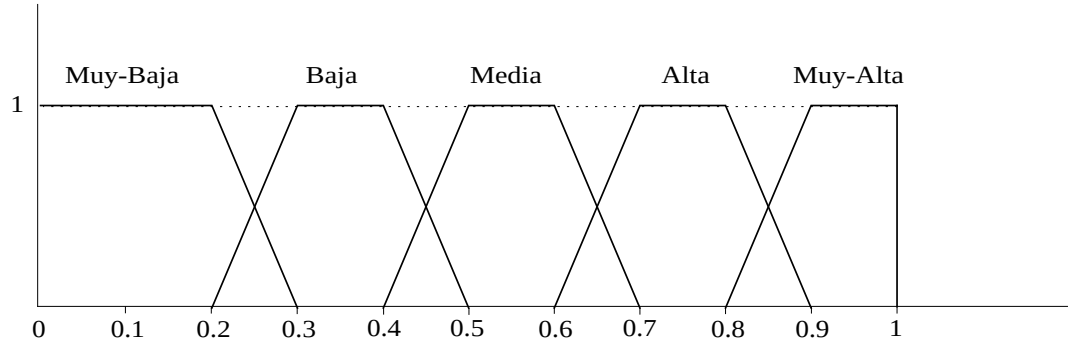


Figura 5.11. Etiquetas correspondientes a la proporcion de alumnos

proporción se considera *Muy-Baja*, *Baja*, *Media*, *Alta* o *Muy-Alta*. La definición de todas estas etiquetas puede verse la figura 5.11

La especificación en Prolog de los enunciados o reglas que hemos descrito quedaría, de forma simplificada, como sigue:

1. *Definición del predicado* `alumno_bueno`.

```
alumno_bueno(Nombre,Asig,Pos,Prob):-
                    alumno(Nombre,Edad,_),
                    matriculado(Nombre,_,Asig,Nota,Actitud),
                    adecuado(Curso,EdadX),
                    comp(edad,EdadX,Edad,Pos1),
                    comp(acti,Actitud,"Positiva",Pos2),
                    mayor_ig(nota,Nota,"Notable",Pos3),
                    min(Pos1,Pos2,M), min(M,Pos3,Pos).
```

La finalidad de los predicados involucrados en esta definición es la siguiente:

- Los predicados `alumno(Nombre,Edad,_)` y `matriculado (Nombre ,_,Asig, Nota,Actitud)` se utilizan para conocer el nombre del alumno, su edad, la calificación que tiene en la asignatura por la que se pregunta y su actitud en dicha asignatura.
- Los predicados `adecuado(Curso,EdadX)` y `comp(edad,EdadX,Edad,Pos1)` se utilizan para conocer cuál es la edad adecuada al curso que dicho alumno está realizando y para compararla con la edad del alumno en cuestión, respectivamente. El grado de compatibilidad entre estas dos edades se almacena en `Pos1`.
- El predicado `comp(acti,Actitud,"Positiva",Pos2)` lleva a cabo la comparación entre la actitud del alumno y una actitud positiva. El grado de semejanza entre ambos valores se almacena en `Pos2`.

- El predicado `mayor_ig(nota,Nota,"Notable",Pos3)` calcula hasta qué punto la nota del alumno considerado supera o iguala el notable. El resultado se guarda en `Pos3`.
- Por último, con los predicados `min(Pos1,Pos2,M)` y `min(M,Pos3,Pos)` se calcula el valor de certeza `Pos` asociado al resultado, utilizando el mínimo como T-Norma.

Sobre los valores `Pos1`, `Pos2` y `Pos3` se pueden exigir umbrales de cumplimiento. Como ninguno de los predicados involucrados lleva asociado una medida de probabilidad, se considera que el valor de `Prob` tras aplicar la regla es 1 (los predicados utilizados han sido aplicados sobre hechos).

2. *Definición del predicado* `curso_dificil`.

curso_dificil(Curso,Pos,Prob):-

num_asig(Curso,Na),

Na≥4,

num_cred(Curso,Nc),

comp(cred,Nc,"Alto",Pos1),

CD=0.8, min(CD,Pos1,Pos).

La finalidad de los predicados involucrados en esta definición es la siguiente:

- Con los predicados `num_asig(Curso,Na)` y `Na≥4` se calcula el número total de asignaturas del curso `Curso` y se exige que éste sea mayor o igual a 4. Definiendo las etiquetas correspondientes, es casi directo hacer esta misma comparación en términos difusos, esto es, exigiendo que el resultado de `comp(asig,Na,"Alto",PosX)` supere un umbral determinado. No se ha hecho así por simplicidad, ya que el caso es completamente análogo al del número de créditos por asignatura.
- Los predicados `num_cred(Curso,Nc)` y `comp(cred,Nc,"Alto",Pos1)` calculan el número total de créditos del curso actual y lo compara con "Alto". El resultado de esta comparación se guarda en `Pos1`.

- Por último, `CD=0.8` es la *"bondad"* de la definición, que se combina con los resultados de los predicados anteriores para calcular `Pos`.

3. *Definición del predicado* `asignatura_mala`.

```
asignatura_mala(Asig,Pos,Prob):-
                    asignatura(Asig,_,Curso),
                    curso_dificil(Curso,Pos1,Prob1),
                    prop_alum_buenos(Asig,Pab),
                    menor_ig(prop,Pab,"Bajo",Pos2),
                    num_alu(Asig,Na), Na>50,
                    Prob=0.9,
                    min(Pos1,Pos2,Pos).
```

La finalidad de los predicados involucrados en esta definición es la siguiente:

- Con el predicado `asignatura(Asig,_,Curso)` se averigua a qué curso corresponde la asignatura.
- Con el predicado `curso_dificil(Curso,Pos1,Prob1)` se averigua hasta qué punto dicha asignatura puede considerarse difícil, utilizando la regla expresamente definida para ello.
- Con los predicados `prop_alum_buenos (Asig,Pab)` y `menor_ig (prop,Pab, "Bajo",Pos2)` se calcula la proporción de alumnos buenos de entre los matriculados en la asignatura `Asig` y se compara esta proporción con la etiqueta `Bajo`. El resultado de dicha comparación se almacena en `Pos2`.
- Con `num_alu(Asig,Na)` y `Na>50` se calcula el número de alumnos matriculados en la asignatura `Asig` y se exige que supere 50. Igualmente se podrían haber definido etiquetas para el número de alumnos por asignatura.
- Por último, `Prob=0.9` establece la probabilidad que tiene asociada la regla y `min(Pos1,Pos2,Pos)` determina el valor de `Pos`.

5.4.1 Ejemplos de Consultas

Veamos los resultados obtenidos tras la realización de algunas consultas sobre las tablas de la BD.

– *Obtener una lista de los alumnos buenos en la asignatura bases de datos.*

```
alumno_bueno(Nombre,"BD",Pos,Prob).
```

Nombre	Pos	Prob
Galvez	1	1
Diego	0.5	1
Heras	0.7	1
Castro	0.7	1
Abad	1	1
Sanchez	1	1

– *Obtener una lista de los cursos junto con su dificultad.*

```
curso_dificil(Curso,Pos,Prob).
```

Curso	Pos	Prob
"1"	0.25	1
"2"	1	1
"3"	0.75	1

– *Obtener un listado de las asignaturas peores de impartir.*

```
asignatura_mala(Asig,Pos,Prob).
```

Asignatura	Pos	Prob
Arquitectura	0.75	0.9
I.Artificial	0.75	0.9
BD	0.75	0.9
Estadistica	0.8	0.9
Calculo	0.8	0.9
S.Logico	0.8	0.9
S.Digitales	0.8	0.9
Electronica	0.8	0.9
PO2	0.8	0.9
Fisica	0.25	0.9
Informatica	0.25	0.9
Analisis	0.25	0.9
Algebra	0.25	0.9
PO1	0.25	0.9

Las **etiquetas** *que se han definido son las siguientes*:

- Para la edad:

  lab(edad,"Muy-Joven",17,17,18,19).

  lab(edad,"Joven",17,19,21,23).

  lab(edad,"Media",21,23,25,27).

  lab(edad,"Mayor",25,27,29,31).

  lab(edad,"Muy-Mayor",29,31,70,70).

  lab(edad,"17-19",17,17,19,19).

  lab(edad,"Apx21",19,21,21,23).

- Para la nota:

  lab(nota,"Muy-Def",0,0,2,3).

  lab(nota,"Def",2,3,4,5).

lab(nota,"Apto",4,5,6,7).

lab(nota,"Notable",6,7,8,9).

lab(nota,"Sobresal",8,9,10,10).

- Para las edades adecuadas a cada curso:

  lab(edad,"Edad1",17,17,19,20).

  lab(edad,"Edad2",18,18,20,22).

  lab(edad,"Edad3",19,19,21,23).

- Para la proporción de alumnos:

  lab(prop,"Muy-Baja",0,0,0.2,0.3).

  lab(prop,"Baja",0.2,0.3,0.4,0.5).

  lab(prop,"Media",0.4,0.5,0.6,0.7).

  lab(prop,"Alta",0.6,0.7,0.8,0.9).

  lab(prop,"Muy-Alta",0.8,0.9,1,1).

- Para el número de créditos:

  lab(cred,"Bajo",1,1,22,26).

  lab(cred,"Medio",22,26,30,34).

  lab(cred,"Alto",30,34,80,80).

Las **relaciones de similitud** *que se han considerado son*:

sim(acti,"Muy-Negativa","Negativa",0.7).

sim(acti,"Negativa","Normal",0.4).

sim(acti,"Normal","Positiva",0.7).

sim(acti,"Normal","Muy-Positiva",0.5).

sim(acti,"Positiva","Muy-Positiva",0.8).

correspondientes a la actitud.

Las siguientes tablas, muestran la base de hechos que se ha utilizado a lo largo de todo el ejemplo.

ALUMNO

Nombre	**Edad**	**Provincia**
Abad	20	Almeria
Alonso	18	Jaen
Amos	21	Granada
Blanco	Mayor	Granada
Blasco	[17-19]	Granada
Castro	19	Almeria
Contreras	Mayor	Malaga
Delgado	Joven	Almeria
Diaz	20	Jaen
Diego	Media	Almeria
Galvez	Joven	Granada
Gamero	19	Toledo
Garcia	Muy-Joven	Jaen
Gomez	Joven	Malaga
Haro	22	Granada
Heras	21	Granada
Huete	Media	Granada
Jimenez	18	Jaen
Larte	Media	Granada
Lijo	Mayor	Jaen
Lopez	Muy-Mayor	Jaen
Mariscal	17	Granada

ALUMNO (Cont.)

Nombre	**Edad**	**Provincia**
Martin	Muy-Joven	Granada
Mateos	20	Malaga
Mayor	23	Almeria
Medina	Media	Almeria
Meras	Media	Jaen
Montes	21	Cadiz
Montiel	18	Jaen
Mora	24	Granada
Moreno	Joven	Granada
Narra	18	Almeria
Pelaez	Joven	Granada
Perez	18	Granada
Pinto	Muy-Joven	Granada
Ramos	Joven	Granada
Romero	Media	Malaga
Saez	23	Granada
Salas	19	Jaen
Samos	Joven	Malaga
Sanchez	19	Granada
Santos	20	Jaen
Segura	Mayor	Almeria
Soria	Aprox. 21	Jaen
Soriano	Muy-Joven	Jaen
Soto	Joven	Malaga
Trujillo	Muy-Mayor	Almeria

Tabla 5.4. Relación de ALUMNOS

ASIGNATURA

Asignatura	Creditos	Curso
PO1	10	1
Algebra	6	1
Analisis	4	1
Informatica	8	1
Fisica	3	1
PO2	9	2
Electronica	6	2
S. Digitales	5	2
S. Logico	7	2
Calculo	5	2
Estadistica	4	2
BD	10	3
I. Artificial	8	3
Arquitectura	8	3
I. Software	7	3

Tabla 5.5. Relación de ASIGNATURAS

MATRICULADO

Nombre	Grupo	Asignatura	Nota	Actitud
Abad	A	BD	Sobresal	Positiva
Abad	A	Arquitectura	Sobresal	Positiva
Abad	A	I.Artificial	4.5	Muy-Negativa
Abad	A	I.Software	3	Negativa
Alonso	A	BD	Def	Negativa
Alonso	A	I.Artificial	Muy-Def	Negativa
Alonso	A	Arquitectura	3	Negativa
Alonso	A	I.Software	4.5	Muy-Negativa
Amos	A	Arquitectura	Apto	Normal
Amos	A	BD	3	Negativa
Amos	A	I.Software	4.5	Muy-Negativa
Amos	A	I.Artificial	Def	Negativa
Blanco	A	I. Artificial	Sobresal	Muy-Negativa
Blanco	A	BD	10	Muy-Positiva
Blanco	A	Arquitectura	Notable	Negativa
Blanco	A	I. de Software	9	Normal
Blasco	A	Informatica	Apto	Negativa
Blasco	A	Algebra	6	Positiva
Blasco	A	PO1	7	Normal
Blasco	A	Analisis	8	Positiva
Castro	A	BD	7.5	Normal
Castro	A	I.Software	Apto	Normal
Castro	A	Arquitectura	Sobresal	Positiva
Castro	A	I.Artificial	Def	Negativa
Contreras	A	PO1	Muy-Def	Negativa
Contreras	A	Algebra	Notable	Positiva
Delgado	A	Fisica	7.5	Normal
Delgado	A	Informatica	Muy-Def	Normal
Diaz	A	I. Artificial	9,5	Negativa
Diaz	A	BD	9	Normal

MATRICULADO (Cont.)

Nombre	Grupo	Asignatura	Nota	Actitud
Diego	A	Arquitectura	7	Positiva
Diego	A	I.Artificial	2	Normal
Diego	A	I.Software	Apto	Normal
Diego	A	BD	7.5	Normal
Galvez	A	Arquitectura	Sobresal	Positiva
Galvez	A	I.Software	4	Normal
Galvez	A	BD	Sobresal	Positiva
Galvez	A	I.Artificial	4	Normal
Gamero	B	Analisis	3.5	Normal
Gamero	B	Algebra	7	Negativa
Gamero	B	Fisica	Muy-Def	Negativa
Gamero	B	Informatica	Def	Normal
Gamero	B	PO1	8	Positiva
Garcia	A	S.Logico	3.5	Positiva
Garcia	A	Electronica	6	Normal
Garcia	A	S.Digitales	1	Negativa
Garcia	A	PO2	Muy-Def	Negativa
Gomez	B	PO1	10	Muy-Positiva
Gomez	B	Informatica	Sobresal	Normal
Haro	B	BD	3	Negativa
Haro	B	Arquitectura	Def	Negativa
Haro	B	I.Software	4.5	Muy-Negativa
Haro	B	I.Artificial	Apto	Normal
Heras	B	Arquitectura	Sobresal	Positiva
Heras	B	I.Software	Sobresal	Positiva
Heras	B	BD	7.5	Normal
Heras	B	I.Artificial	4.5	Muy-Negativa
Huete	B	Analisis	Def	Normal
Huete	B	Informatica	Notable	Positiva
Huete	B	PO1	Notable	Positiva

MATRICULADO (Cont.)

Nombre	Grupo	Asignatura	Nota	Actitud
Huete	B	Fisica	3.5	Positiva
Huete	B	Algebra	3.5	Positiva
Jimenez	A	Estadistica	2.5	Normal
Jimenez	A	Calculo	7.5	Normal
Larte	B	Arquitectura	2	Normal
Larte	B	I.Artificial	4	Normal
Larte	B	BD	Muy-Def	Negativa
Larte	B	I.Software	Sobresal	Positiva
Lijo	B	S.Digitales	Notable	Positiva
Lijo	B	Electronica	4	Normal
Lijo	B	PO2	Muy-Def	Negativa
Lijo	B	Calculo	Apto	Normal
Lijo	B	Estadistica	7.5	Normal
Lijo	B	S.Logico	7.5	Normal
Lopez	B	I.Software	4	Normal
Lopez	B	BD	Sobresal	Positiva
Lopez	B	Arquitectura	7.5	Normal
Lopez	B	I.Artificial	Muy-Def	Negativa
Mariscal	B	Estadistica	4.5	Positiva
Mariscal	B	PO2	4	Normal
Mariscal	B	Electronica	6	Normal
Martin	B	Analisis	Notable	Normal
Mateos	B	Analisis	9	Muy-Positiva
Mateos	B	Fisica	8.5	Normal
Mateos	B	Informatica	10	Positiva
Mayor	B	S.Logico	6	Normal
Mayor	B	Calculo	7.5	Normal
Medina	B	BD	Notable	Normal
Meras	B	Calculo	Notable	Positiva
Meras	B	Estadistica	Muy-Def	Negativa

MATRICULADO (Cont.)

Nombre	Grupo	Asignatura	Nota	Actitud
Meras	B	S.Digitales	Notable	Positiva
Meras	B	S.Logico	Apto	Normal
Montes	B	Algebra	3.5	Positiva
Montes	B	Analisis	Def	Normal
Montes	B	PO1	Muy-Def	Negativa
Montiel	B	S.Digitales	7.5	Normal
Montiel	B	Electronica	6	Normal
Montiel	C	S.Logico	Muy-Def	Negativa
Montiel	B	PO2	Def	Normal
Mora	B	S. Logico	10	Normal
Mora	B	S. Digitales	Sobresal	Positiva
Mora	B	Electronica	Apto	Normal
Mora	B	PO2	Notable	Normal
Moreno	B	PO1	3.5	Positiva
Moreno	B	Informatica	5	Normal
Moreno	B	Analisis	Def	Muy-Positiva
Moreno	B	Algebra	Muy-Def	Negativa
Narra	B	Calculo	4	Normal
Narra	B	S.Logico	Muy-Def	Negativa
Narra	B	S.Digitales	7.5	Normal
Pelaez	C	PO2	Apto	Normal
Pelaez	C	S.Digitales	Def	Normal
Pelaez	C	Estadistica	7.5	Normal
Pelaez	C	Electronica	Notable	Positiva
Perez	C	Analisis	Apto	Muy-Negativa
Perez	C	PO1	7	Positiva
Perez	C	Algebra	Sobresal	Negativa
Pinto	C	Analisis	Sobresal	Muy-Positiva
Pinto	C	PO1	Apto	Normal
Pinto	C	Fisica	7	Normal

MATRICULADO (Cont.)

Nombre	Grupo	Asignatura	Nota	Actitud
Pinto	C	Algebra	Notable	Positiva
Pinto	C	Informatica	Apto	Negativa
Ramos	C	Arquitectura	7.5	Normal
Ramos	C	I.Software	Muy-Def	Negativa
Ramos	C	BD	4	Normal
Ramos	C	I.Artificial	Apto	Normal
Romero	C	Algebra	4	Normal
Romero	C	Informatica	Notable	Positiva
Romero	C	Analisis	Apto	Normal
Romero	C	PO1	6	Positiva
Romero	C	Fisica	2	Negativa
Saez	C	Electronica	Muy-Def	Negativa
Saez	C	S.Digitales	4	Normal
Saez	C	PO2	4	Normal
Salas	D	PO2	Apto	Normal
Salas	D	S. Digitales	2	Negativa
Salas	D	S. Logico	Def	Normal
Salas	D	Electronica	4	Positiva
Samos	C	I.Artificial	4	Normal
Samos	C	Arquitectura	Muy-Def	Negativa
Samos	C	BD	Notable	Positiva
Samos	C	I.Software	Sobresal	Positiva
Sanchez	C	I.Artificial	6.5	Normal
Sanchez	C	BD	Sobresal	Positiva
Segura	C	S.Logico	7.5	Normal
Segura	C	Calculo	4	Normal
Segura	C	Estadistica	2.5	Normal
Soria	C	I. Artificial	Def	Negativa
Soria	C	BD	Apto	Muy-Negativa
Soria	C	Arquitectura	Muy-Def	Muy-Negativa

MATRICULADO (Cont.)

Nombre	**Grupo**	**Asignatura**	**Nota**	**Actitud**
Soriano	C	I.Artificial	7.5	Normal
Soriano	C	Arquitectura	4	Normal
Soriano	C	I.Software	Muy-Def	Negativa
Soriano	C	BD	4	Normal
Soto	C	Calculo	Muy-Def	Negativa
Soto	C	Electronica	4	Normal
Soto	C	Estadistica	Apto	Normal
Soto	C	S.Logico	4	Normal
Soto	C	PO2	7.5	Normal
Trujillo	C	S.Digitales	6	Normal
Trujillo	C	Electronica	4	Normal
Trujillo	C	PO2	6	Positiva
Trujillo	C	Estadistica	3.5	Positiva
Trujillo	C	Calculo	4	Normal

Tabla 5.6. Relación de MATRICULAS

Conclusiones y Trabajos Futuros

- **Resultados y Conclusiones**

  Los resultados y conclusiones más interesantes que pueden obtenerse tras la realización de este trabajo, pueden resumirse en los siguientes puntos, como sigue:

  - *Respecto al modelo lógico de bases de datos difusas introducido:*
    - Permite operar con una amplia gama de tipos de datos difusos, de hecho, con todos los tipos que se presentan en los modelos de la literatura.
    - Permite representar completamente (tanto en lo que se refiere a los datos como a su manipulación) los principales modelos de bases de datos difusas de la literatura.
    - El modelo presenta gran flexibilidad para el tratamiento y evaluación de información difusa y gran homogeneidad en la representación tanto de los datos difusos como de los precisos.
    - Posee grandes posibilidades de ampliación en lo que se refiere a los operadores utilizados, las etiquetas, las restricciones,...
  - *Respecto al modelo de deducción propuesto:*
    - Está basado en el concepto de *"compatibilidad"* o *"adecuación"* entre consultas y respuestas, concepto que resulta bastante natural de cara a usuarios no familiarizados con los conceptos de conjunto difuso, distribución de posibilidad, relación de semejanza, etc...
    - Respeta el modus ponens clásico y presenta un buen comportamiento en lo que al modus ponens difuso se refiere, produciendo respuestas cada vez más *"amplias"* (inciertas) conforme decrece la compatibilidad con el antecedente o cuánto peor es la regla aplicada.

  - Permite ampliar tanto el esquema como la información contenida en la base de datos a partir del esquema previo y de la información contenida en la misma.

- *Respecto a la implementación llevada a cabo:*
  - Los algoritmos de conversión de datos (de tablas a predicados) resultan rápidos y eficientes.
  - Resulta inmediata la inserción de nuevos operadores, comparadores y etiquetas.
  - Dado que el volumen de hechos no será, en la mayoría de los casos, excesivo (sólo se maneja información parcial de la base de datos real) los tiempos de respuesta y deducción son más que aceptables. A ello contribuye enormemente el hecho de haber indexado los predicados y las etiquetas utilizadas así como la utilización eficiente del *backtraking* y el corte.

En resumen, hemos probado que se pueden aprovechar las ventajas y las posibilidades que ofrecen tanto los sistemas de gestión de bases de datos relacionales como la lógica difusa de forma conjunta, en la construcción de un modelo general de base de datos lógica difusa con capacidad de deducción. Hemos visto, además, que el lenguaje Prolog y su entorno ofrece un marco de trabajo adecuado a nuestro propoósito, aunque con alguna limitación (no está especializado en el tratamiento de grandes volúmenes de datos). De hecho, en ningún momento se pretende sustituir un DBMS comercial por un sistema lógico como el presentado, ya que los primeros contienen, por lo general, mecanismos muy complejos y altamente especializados en la manipulación de grandes cantidades de datos, como generadores de índices, gestores de memoria, optimizadores de consultas, mecanismos eficientes de ordenación, generadores de informes,..... que un sistema de programación lógica, por bueno que éste sea, no contempla.

- **Trabajos Futuros**

Dentro de las líneas de trabajo que pueden quedar abiertas, se encuentran opciones relacionadas tanto con aspectos teóricos como de implementación. Estos puntos pueden resumirse como sigue:

* Realizar un estudio teórico y la posterior implementación de un operador de negación adecuado a la semántica adoptada a las proposiciones cualificadas posibilísticamente. Esta negación ha de definirse de manera particular dado que una proposición cualificada posibilísticamente hace referencia a un conjunto de resultados posibles a cierto grado.
* Ampliar el modelo multivaluado de valoración de compatibilidad y probabilidad a un modelo *difuso*, en el que se manipulen etiquetas tales como poco compatible, escasamente compatible, medianamente compatible, ... y poco probable, escasamente probable, muy probable,.... Respecto a las distintas formas de hacerlo puede verse [77].
* Implementar completamente el módulo de control que interactúa con el SGBD y traspasa información de un entorno a otro.
* Extender la filosofía de los modelos estudiados a problemas de representación y manipulación de datos más estructurados, como los que aparecen en la representación por marcos o "frames" y en las bases de datos orientadas a objetos.

Bibliografía

[1] : Alsina C.,Trillas E.,Valverde L.,1983. *On some Logical Connectives for Fuzzy Set Theory.* Journal of Math. Anal. and Appl., Vol. 93, 15-26.

[2] : Anvari M.,Rose G.F.,1987. *Fuzzy Relational Databases.* Analysis of Fuzzy Information. Bezdek Eds. Vol. II CRC Press.

[3] : Baldwin J.F.,1978. *A New Approach to Approximate Reasoning Using a Fuzzy Logic.* Fuzzy Sets and Systems 2, 309-325.

[4] : Baldwin J.F., Guild N.,1979. *Comparison of Fuzzy Sets on the Same Decision Space.* Fuzzy Sets and Systems 2, Pp. 213-233.

[5] : Baldwin J.F., Zhou S.Q.,1984. *A Fuzzy Relational Inference Language.* Fuzzy Sets and Systems 14, Pp. 155-174. North-Holland.

[6] : Baldwin J.F.,1987. *FRIL-A Fuzzy Relational Inference Language.* Fuzzy Sets and Systems 14, Pp. 155-174.

[7] : Bandler W.,Kohout L.J.,1979. *Semantics of Implication Operators and Fuzzy Relational Products.* En Fuzzy Reasoning and its Applications. Pp. 216-246. Computers and People Series. Eds. Mamdani y Gaines. Academic Press. Londres.

[8] : Bocca J.,1986. *On the Evaluation Strategy of EDUCE.* Proc de ACM-SIGMOD. Washington.

[9] : Bocca J.,Decker H.,Nicolas J.M.,1986. *Some Steps towards a DBMS-Based KBMS.* Proc. de la Conf. Int. IFIP. Dublin.

[10] : Brodie M., Jarke M.,1986. *On Integrating Logic Programming and Databases.* En Expert Database Systems, Ed. Larry Kerschberg.

[11] : Buchanan B.B.,Shortliffe E.E.,1984. *Rule-Based Expert Systems- The MYCIN Experiments of the Stanford Heuristic Programming Project.* Addison-Wesley Reading.

[12] : Buckles B.P.,Petry F.E.,1982. *A Fuzzy Representation of Data for Relational Databases.* Fuzzy Sets and Systems, Vol. 7, Pp. 213-236.

[13] : Buckles B.P.,Petry F.E.,1984. *Extending the Fuzzy Database Model with Fuzzy Numbers.* Information Sciences 34, Pp. 145-155.

[14] : Buckles B.P.,Petry F.E.,Sachar H.S.,1989. *A Domain Calculus for Fuzzy Relational Databases.* Fuzzy Sets and Systems, Vol. 29, Pp. 327-340.

[15] : Castro J.L.,Trillas E.,1993. *The Management of the Inconsistency in Expert Systems.* Fuzzy Sets and Systems, Vol. 58, Pp. 51-57.

[16] : Ceri S.,Gottlob G., Wiederhold G.,1986. *Interfacing Relational Databases and Prolog Efficiently.* En Expert and Database Systems. Ed. Kerschberg. Benjamin-Cummings.

[17] : Ceri S.,Gozzi F.,Lugli M.,1988. *An Overview of PRIMO: A Portable Interface between Prolog and Relational Databases.* Tech. Report CS School. Univ. Modena.

[18] : Ceri S.,Gottlob G., Wiederhold G.,1989. *Efficient Database Access through Prolog.* IEEE Transactions on Software Engineering.

[19] : Ceri S.,Gottlob G.,Tanca L.,1990. *Logic Programming and Databases.* Pp. 29-38. Ed. Springer-Verlag.

[20] : Ceri S.,Gottlob G.,Tanca L.,1990. *Logic Programming and Databases.* Pp. 47-64. Ed. Springer-Verlag.

[21] : Chang C.L.,Walker A.,1984. *PROSQL: A Prolog Programming Interface with SQL/DS.* Proc. del 1^{er} Workshop en Expert Database Systems, Kiawah (Islandia).

[22] : Clocksin S.F., Mellish C.S.,1981. *Programming in Prolog.* Ed. Springer-Verlag.

[23] : Codd E.F.,1970. *A Relational Model of Data for Large Shared Data Banks.* Com. ACM, 13(6) Pp. 377-387.

[24] : Codd E.F.,1971. *A Data Sublanguage Founded on the Relational Calculus.* Eds. Codd y Dean. Proc. de ACM SIGFIDET Workshop on Data Description, Access and Control. San Diego.

[25] : Codd E.F.,1979. *Extending the Database Relational Model to Capture More Meaning.* ACM Trans. on Database Systems.

[26] : Codd E.F.,1982. *Relational Databases: A Practical Foundation for Productivity.* Com. ACM, 25(2).

[27] : Codd E.F.,1985. *How Relational Is Your Database Management System.* Computerworld, Octubre 14, 21.

[28] : Codd E.F.,1987. *More Commentary on Missing Information in Relational Databases.* ACM SIGMOD Record, Vol. 16, 1.

[29] : Codd E.F.,1986. *Missing Information (Applicable and Inapplicable) in Relational Databases.* ACM SIGMOD Record, Vol. 15, 4.

[30] : Codd E.F.,1986. *The Twelve Rules for Relational DBMS.* The Relational Institute, Technical Report EFC-6. San Jose.

[31] : Codd E.F.,1990. *The Relational Model for Database Management. Version 2.* Reading. Addison Wesley.

[32] : Cubero J.C., Diaz J., Medina J.M., Pons O., Prados M., Vila M.A., 1991. *Un Modelo de Base de Datos Difusa Aplicado a Información Médica.* Com. del I Congreso Español sobre Tecnologías y Lógica Fuzzy.

[33] : Cubero J.C., Diaz J., Medina J.M., Pons O., Prados M., Vila M.A., 1991. *Un Modelo de Representación del Conocimiento para Bases de Datos Imprecisas.* Asociación Española Para la Inteligencia Artificial (AEPIA). Madrid.

[34] : Cuppens F.,Demolombe R.,1986. *A Prolog- Relational DBMS Interface using Delayed Evaluation.* Workshop sobre Integración de la Programación Lógica y Bases de Datos. Venecia.

[35] : Date C.J.,1985. *An Introduction to Database Systems, Vol. II.* The Systems Progamming Series. Addison Wesley.

[36] : Date C.J.,1990. *An Introduction to Database Systems.* Vols. I,II. Addison Wesley.

[37] : Deaño A.,1974. *Introducción a la Lógica Formal.* Ed. Alianza Universidad.

[38] : de Campos L.M., Gonzalez A.,1993. *A Fuzzy Inference Model Based on an Uncertainty Forward Propagation Approach.* Int. Journal of Approximate Reasoning, Vol. 9, Pp. 139-164. Elsevier Science Publishing Co.

[39] : DeGroot D., Lindstrom G.,1986. *Logic Programming. Functions, Relations and Equations.* Ed. Prentice Hall, Englewood Cliffs, New Jersey.

[40] : Delgado M.,Verdegay J.L.,Vila M.A.,1993. *Decision Making Models.* Por aparecer en International Journal of Intelligent Systems.

[41] : Cholvy L., Demolombe R.,1986. *Querying a Rule Base.* En Proc. de 1st. Int. Conference on Expert Databse Systems.

[42] : Demolombe R.,1992. *A Strategy for the Computation of Conditional Answers.* En 10th European Conference on Artificial Intelligence.

[43] : Demolombe R.,1992. *Uncertainty in Intelligent Data Bases.* Tech. Rep. ONERA/CERT. Tolouse.

[44] : Demolombe R.,1992. *Cooperative Answering from Computerised Databases.* Tech. Rep. ONERAL/CERT. Tolouse.

[45] : Denoel D. et al.,1986. *Query Transltation for Coupling Prolog with a Relational Database Management System.* Workshop sobre Integración de de la Programación Lógica y Bases de Datos. Venecia.

[46] : Dubois D.,Prade H.,1980. *Fuzzy Sets and Systems: Theory and Applications.* Academic Press. New York.

[47] : Dubois D.,Prade H.,1985. *Combination and Propagation of Uncertainty with Believe Functions.* Proc. of 9th Int. Joint Conf. on Artificial Intelligence. Pp. 111-113. Pub. Morgan-Kaufmann.

[48] : Dubois D.,Prade H.,1990. *Modelling Uncertain and Vague Knowledge in Possibility and Evidence Theories.* En Uncertainty in Artificial Intelligence, Vol. 4, Pp. 303-318.

[49] : Dubois D.,Prade H.,1991. *Fuzzy Sets in Approximate Reasoning. Part I: Inference with Possibility Distributions.* Fuzzy Sets and Systems 40, 143-202.

[50] : Dubois D.,Lang J.,Prade H.,1991. *Fuzzy Sets in Approximate Reasoning. Part II: Logical Approaches.* Fuzzy Sets and Systems 40, 203-.

[51] : Fukami S., Umano M., Muzimoto M., Tanaka H.,1979. *Fuzzy Database Retrieval and Manipulation Language.* IEICE Technical Reports Vol. 78, Num. 233, Pp. 65-72.

[52] : Fukami M.,Mizumoto M.,Tanaka K.,1980. *Some Considerations of Fuzzy Conditional Inference.* Fuzzy Sets and Systems 4, Pp. 243-273.

[53] : Frost R.A.,1986. *Introduction to Knowledge Base Systems.* Ed. Collins.

[54] : Gaines B.R.,1978. *Logical Foundations for Database Systems.* En Fuzzy Reasoning and its Applications. Pp. 289-308. Computers and People Series. Eds. Mamdani y Gaines. Academic Press. Londres.

[55] : Gal A.,Minker J.,1985. *A Natural Language Inteface that Provides Cooperative Answers.* En Proc. 2º Conference on Artificial Intelligence Applications.

[56] : Gal A.,Minker J.,1990. *Producing Cooperative Answers in Deductive Databases.* En Logic and Logic Grammar for Language Processing, de L.S. Howard. Eds. Saint- Dizier y Szpakowics.

[57] : Gallaire H.,1978. *Logic and Databases.* Eds. Gallaire y Minker. Plenum Press.

[58] : Gallaire H.,Minker J.,Nicolas J.M.,1984. *Logic and Databases: A Deductive Approach.* ACM Computing Surveys. Vol. 16(2), Pp. 153-185.

[59] : Gallaire H.,Nicolas J.M.,1987. *Logic Approaches to Knowledge and Databases.* IEEE Data Engineering. Vol. 10(4). Ed. Zaniolo.

[60] : Godo L., López de Mántaras R., Sierra C., Verdaguer A., 1988. *Managing Linguistically Expressed Uncertainty in MILORD: Application to Medical Diagnosis.* Artificial Intelligence Communications, Vol. 1(1), Pp. 14-31.

[61] : Gray P., 1984. *Representing Programs by Clauses.* En Logic, Algebra and Databases. Pp. 73-96. Ed. Meek B.L. Londres.

[62] : Herken R.,1988. *The Universal Turing Machine.* Ed. Rolf Herken. Oxford University Press.

[63] : Hermes H.,1984. *Introducción a la Teoría de la Computabilidad.* Ed. Tecnos S.A.

[64] : Hogger C.J.,1990. *Essentials of Logic Programming.* Eds. D.M. Gabbay, C.L. Hankin y T.S. Maibaum. Clarendon Press. Oxford.

[65] : Imielinski T., Lipski W.,1984. *Incomplete Information in Relational Databases.* En ACM, Vol. 31(4).

[66] : Ioannidis Y.E. et al.,1987. *BERMUDA: An Architectural Perspective on Interfacing Prolog to a Database Machine.* Tech. Report 723, Computer Science Dep., Wisconsin.

[67] : Jeffrey R.C.,1986. *Lógica Formal. Su alcance y sus límites.* Ed. Universidad de Navarra S.A. Pamplona.

[68] : Kim W., 1990. *Object-Oriented Databases: Definition and Research Directions.* IEEE Trans. On Knowledge and Data Engineering, Vol. 2(3), Pp. 327-341.

[69] : Korth H.F.,Silberschatz A.,1993. *Fundamentos de Bases de Datos.* Ed. Mc-Graw Hill.

[70] : Kowalski R.,1979. *Logic for Problem Solving* . Pp. 35-53. Ed. North Holland Publishing Co.

[71] : Kowalski R.,1979. *Logic for Problem Solving* . Pp. 93-181. Ed. North Holland Publishing Co.

[72] : Kowalski R.,1981. *Logic as a Database Languaje.* Seminario sobre los Aspectos Teóricos de las BD. Italia.

[73] : Larsen H.L.,Nonfjall H.N.,Bouteldja N.,Yager R.R.,1989. *Checking a Knowledge Base for Potential Inconsistency, Part II.* Tech. Report (ESPRIT II-2148).

[74] : Lee K.M., Lee-Kwang H.,1992. *Fuzzy Matching and Fuzzy Comparison in Fuzzy Expert Systems.* Proc. de II Int. Conf. on Fuzzy Logic and Neural Networks. Pp. 313-316. Iizuka. Japón.

[75] : Li D.,1984. *A Prolog Database System.* Research Institute Press, Letchworth, Hertfordshire, Inglaterra.

[76] : Li D., Liu D., 1990. *A Fuzzy Prolog Database System.* Ed. John Wiley and Sons, New York.

[77] : López de Mántaras R.,1990. *Approximate Reasoning Models.* Ellis Horwood Series in Artificial Intelligence.

[78] : Lorenzen P.,1970. *Lógica Formal.* Selecciones Científicas. Madrid.

[79] : Lucas R.,1988. *Database Applications using Prolog.* Ellis Horwood Series en Computers and their Applications. Londres.

[80] : Lucas R.J.,Le Vine G.A.,1988. *A Prolog-Relational Database Interface.* En Prolog and Databases, Pp. 67-80. Eds. Gray y Lucas, Ellis Horwood. Inglaterra.

[81] : Lunn K.,Archibald G.,1988. *TREQL: An Intelligent Front-End to a Relational Database.* En Prolog and Databases, Pp. 39-51. Eds. Gray y Lucas, Ellis Horwood. Inglaterra.

[82] : Medina J.M., Vila M.A., Cubero J.C., Pons O.,1993. *Towards the Implementation of a Generalized Fuzzy Relational Database Model.* Sometido a Fuzzy Sets and Systems.

[83] : Medina J.M.,Pons O.,Vila M.A.,1993. *GEFRED: A Generalized Model of Fuzzy Relational Databases.* Aceptado en la revista Information Sciences.

[84] : Meseguer P.,1990. *Inconsistency Checking in Rule-based Expert Systems with Uncertainty and Control Features.* ISMIS'90.

[85] : Motro A.,1984. *Query Generalization: A Technique for Handling Query Failure.* En Proc. First International Workshop on Expert Database Systems, Pp. 314-325, Kiawah Island, South Carolina.

[86] : Motro A.,1986. *Integrity = Validity+Completeness.* ACM TODS, 14(4).

[87] : Minker J.,1988. *Foundations of Deductive Databases and Logic Programming.* Ed. Morgan-Kauffman.

[88] : Mosterin J.,1983. *Lógica de Primer Orden.* Ed. Ariel S.A. Barcelona.

[89] : Mouta F.,Williams M.H.,Neves J.M.,1988. *Implementing Query Languages in Prolog.* En Prolog and Databases, Pp. 13-21. Eds. Gray y Lucas.

[90] : Mukaidono M.,Shen Z.,Ding L.,1987. *Fuzzy Prolog*. Preprints of Second IFSA Congress. Tokyo.

[91] : Mukaidono M.,Shen Z.,Ding L.,1988. *A Theoretical Framework of Fuzzy Prolog Machine*. Fuzzy Computing. M.M. Gupta and T. Yamakawa Eds. Pp. 89-100. North-Holland.

[92] : Nakamura A.,Gao J.,1991. *A Logic for Data Analysis*. Fuzzy Sets and Systems 39, Pp. 127-132. North-Holland.

[93] : Nilsson N.J.,1987. *Principios de Inteligencia Artificial*. Ed. Diaz Santos.

[94] : Nussbaum M.,1992. *Building a Deductive Database*. Ablex Pub. Corp. Norwood, New Yersey.

[95] : Piatetsky-Shapiro G., Frawley W.,1991. *Knowledge Discovery in Databases*. Eds. Piatetski-Shapiro and Frawley. California.

[96] : Pons O.,Vila M.A.,1992. *Interface entre BD Relacionales Difusas y el Entorno de Prolog*. En Proc. del II Congreso Español sobre Tecnologías y Lógica Fuzzy. Pp. 215-232. Madrid.

[97] : Pons O, Vila M.A., Delgado M.,1993. *Inferencia a Partir de una BD Difusa Utilizando Reglas Difusas*. Com. del III Congreso Español sobre Tecnologías y Lógica Fuzzy. Santiago de Compostela.

[98] : Pons O.,Vila M.A.,Medina J.M.,1994. *Handling Imprecise Medical Information in the Framework of Logic Fuzzy Databases*. Por aparecer en International Journal on Fuzzy Systems and Artificial Intelligence. Ed. H.N. Teodorescu.

[99] : Prade H.,Testemale C.,1984. *Generalizing Database Relational Algebra for the Treatment of Incomplete or Uncertain Information and Vague Queries*. Information Sciences Vol. 34, Pp. 115-143.

[100] : Prade H.,Testemale C.,1987. *Representation of Soft Constraints and Fuzzy Attribute Values by means of Possibility Distributions in*

Databases. Bezdek Ed. Analysis of Fuzzy Information. Vol II CRC Press.

[101] : Quintus Computer Systems Inc. *Quintus Prolog Database Interface Manual*. Mountain View. California.

[102] : Reiter R.,1984. *Towards a Logical Reconstruction of Relational Database Theory*. En Conceptual Modellig. Eds. Brodie, Mylopoulos y Schmidt. Pp. 193-238.

[103] : Reiter R., 1987. *Nonmonotonic Reasoning*. En Annual Reviews of Computer Science Vol. 2.

[104] : Rundensteiner E.A.,Hawkes L.W.,Bandler W.,1989. *On Nearness Measures in Fuzzy relational Data Models*. Int. Journal of Approx. Reasoning Vol. 3, Pp. 267-298.

[105] : Ruspini E.,1991. *On the Semantics of Fuzzy Logic*. Int. Journal of Approximate Reasoning (IJAR), Vol. 5(1), Pp. 45-88.

[106] : Sacca D.,Zaniolo C.,1986. *On the Implementation of a Simple Class of Logic Queries for Databases*. ACM SIGMOD-SIGACT. Simposium sobre Principios de los Sistemas de Bases de Datos. Cambridge.

[107] : Shen Z.,Ding L.,Mukaidono M.,1988. *A Theoretical Framework of Fuzzy Prolog Machine*. Eds. M.M. Gupta y T. Yamakawa. Pp. 89-100. North Holland.

[108] : Shenoi S.,Melton A.,1989. *Proximity Relations in the Fuzzy Relational Database Model*. Fuzzy Sets and Systems 31, Pp. 285-296.

[109] : Shenoi S.,Melton A.,1990. *An Extended Version of the Fuzzy Relational Database Model*. Information Sciences 52, Pp. 35-52.

[110] : Shmueli O.,Tsur S., Zfira H.,1986. *Rule Support in Prolog*. En Expert Database System, Ed. Larry Kerschberg.

[111] : Sterling C., Shapiro E.,1986. *The Art of Prolog*. Ed. MIT-Press.

[112] : Sugeno M.,1974.*Theory of Fuzzy Integrals and its Applications*. Tesis Doctoral. Instituto de Tecnología. Tokio. Japón.

[113] : Suppes P.,1966. *Probabilistic Inference and the Concept of Total Evidence*. Hintikka and Suppes Eds. Aspects of Inductive Logic, Pp. 49-65. North-Holland.

[114] : Trillas E.,1980. *Conjuntos Borrosos*. Ed. Vicens-Vives.

[115] : Turksen I.B., Zhong Z.,1990. *Analogical Reasoning Approach Based on Similarity Measures*. Fuzzy Sets and Systems 34, Pp. 323-346.

[116] : Ullman F.D.,1989. *Principles of Database and Knowledge-Base Systems*. Computer Science Press. New York.

[117] : Umano M.,Mizumoto M.,Tanaka, 1978. *FSTDS System: A Fuzzy Set Manipulation System*. Information Sciences Vol. 14, Pp. 115-159.

[118] : Umano M.,1982. *FREEDOM-0: A Fuzzy Database System*. EN M.M. Gupta y E. Sanchez Eds. Fuzzy Information and Decision Processes, Pp. 339-347. North Holland Pub. Co.

[119] : Umano M.,1987. *Fuzzy Set Prolog*. Proc. of II Int. Fuzzy Systems Assoc. Pp. 750-753.(IFSA Congress). Tokio (Japón).

[120] : Vila M.A.,Cubero J.C.,Medina J.M.,Pons O.,1992. *A Logical Approach to Fuzzy Relational Databases*. Proc. de IPMU'92.

[121] : Vila M.A.,1992. *Proyecto de Cátedra*.

[122] : Vila M.A.,Cubero J.C.,Medina J.M.,Pons O.,1993. *On the Use of Logical Definition of Fuzzy Relational Database*. Proc. de 2º IEEE Int. Conf. on Fuzzy Systems. Vol. I, Pp. 489-495. San Francisco.

[123] : Vila M.A.,Cubero J.C.,Medina J.M.,Pons O.,1994. *Logic and Fuzzy Relational Databases: A New Language and a New Definition*. Cap. de Fuzzy Sets and Possibility Theory in Databases Management Systems. P. Bosc y J. Kacprzyk. Ed. Physica- Verlag.

[124] : Winston P.H.,1984. *Artificial Intelligence*. Ed. Addison Wesley.

[125] : Yager R.,1982. *Fuzzy Set and Possibility Theory*. Ed. R. Yager. Pergamon Press.

[126] : Zadeh L.A.,1965. *Fuzzy Sets*. Information and Control 8, Pp.338-353.

[127] : Zadeh L.A.,1971. *Similarity Relations and Fuzzy Orderings*. Information Sciences, Vol.3, Pp. 177-200.

[128] : Zadeh L.A.,1975. *The Concept of Linguistic Variable and its Application to Approximate Reasoning I*. Information Sciences, Vol.8, Pp. 199-249.

[129] : Zadeh L.A.,1975. *The Concept of Linguistic Variable and its Application to Approximate Reasoning II*. Information Sciences, Vol.8, Pp. 301-357.

[130] : Zadeh L.A.,1976. *The Concept of Linguistic Variable and its Application to Approximate Reasoning III*. Information Sciences, Vol.9, Pp. 43-80.

[131] : Zadeh L.A.,1977. *PRUF: A Meaning Representation Language for Natural Languages*. En Fuzzy Reasoning and its Applications. Pp. 1-66. Computers and People Series. Eds. Mamdani y Gaines. Academic Press. Londres.

[132] : Zadeh L.A.,1978. *Fuzzy Sets as a Basis for a Theory of Possibility*. Fuzzy Sets and Systems 1, Pp. 3-28.

[133] : Zadeh L.A.,1983. *The Role of Fuzzy Logic in the Management of Uncertainty in Expert Systems*. Fuzzy Sets and Systems Vol. 11, Pp. 199-227.

[134] : Zadeh L.A.,1983. *A Computational Approach to Fuzzy Quantifiers in Natural Languages*. Comp. and Maths. with Appls., Vol. 9, Pp. 149-184.

[135] : Zadeh L.A.,1986. *Outline of a Computational Approach to Meaning and Knowledge Representation Based on the Concept of Generalized Assignment Statement.* Lecture Notes in Control and Inf. Sciences, Pp. 199-210. Eds. M. Thoma y A. Wyner. Springer-Verlag.

[136] : Zadeh L.A.,1987. *A Computational Theory of Dispositions.* Int. Journal of Intelligent Systems, Vol. 2, Pp. 39-63.

[137] : Zaniolo C.,1984. *Prolog: A Database Query Language for All Seasons.* Proc. First Workshop on Expert Database Systems. Kiawah. Island.

[138] : Zemankova-Leech M.,Kandel A.,1984. *Fuzzy Relational Databases: A Key for Expert Systems.* Ed. Verlag TUV Rheiland.

[139] : Zemankova-Leech M.,Kandel A.,1985. *Implementing Imprecision in Information Systems.* Information Sciences, Vol. 37, Pp. 107-141.

[140] : Zwick R.,Carlstein E.,Budescu D.V.,1987. *Measures of Similarity Among Fuzzy Concepts: A Comparative Analysis.* Int. Journal of Approximate Reasoning, Pp. 221-242.