Interfaz gráfica para el manejo base de datos deductivas difusas

Norma Verónica Ramírez Pérez
Instituto Tecnológico de Celaya
norma.ramirez@itcelaya.edu.mx

Julio Armando Asato España
Instituto Tecnológico de Celaya
julio.asato@itcelaya.edu.mx

Carlos Gerardo Euresty Uribe
Instituto Tecnológico de Celaya
gerardo.euresty@itcelaya.edu.mx

Martin Laguna Estrada
Instituto Tecnológico de Celaya
martin.laguna@itcelaya.edu.mx

Marleth Mendoza Jiménez
Instituto Tecnológico de Celaya
marleth412@hotmail.com

Resumen

En este trabajo se propone un sistema de base de datos deductiva difusa con métodos certeros lógicos y difusos para formular consultas basada en Sistemas gestores de base de datos relacionales por medio de una interfaz gráfica. Para esto se adaptaron técnicas precedentes de programación lógica basada con lógica difusa que permite combinar en un mismo contexto etiquetas con el objeto de tener respuestas más acertadas al momento de

consultar y conocer información más confiable del sistema gestor de base de datos.

Palabra(s) Clave(s): Bases de Datos, Interfaz gráfica, Lógica difusa, Programación lógica.

1. Introducción

Los Sistemas Gestores de Base Datos Relacionales (SGBDR) se utilizan para almacenar los datos de un gran número de empresas y organizaciones, Su consulta se realiza empleando un lenguaje especifico siendo el más conocido el Lenguaje Estructurado de Consulta (SQL). En la mayoría de ocasiones, los usuarios no están familiarizados con el almacenamiento de los datos y un experto se encarga de acceder a ellos, traduciendo las necesidades del usuario (consultas expresadas en lenguaje natural) a la sintaxis del lenguaje de consulta. Para que la consulta sea eficaz el experto debe dominar una serie de estrictos comandos lo que unido a un modelo de recuperación rígido podría causar la perdida de algunas respuestas de interés.

Por otro lado, los SGBDR se crearon, en un principio, para trabajar con datos bancarios. Sin embargo las nuevas necesidades de las empresas y organizaciones requieren, además de almacenar y consultar datos, obtener información relevante que les permita obtener conocimiento para tomar decisiones en base a él. Como la mayoría de este conocimiento se expresa en lenguaje natural, es importante que los sistemas de base de datos faciliten, a los expertos de un área, el acceso a los datos de manera sencilla, amigable y haciendo uso de interfaces de consulta flexibles y cercanas. A este respecto, las técnicas basadas en la teoría de conjuntos difusos han resultado muy útiles para crear un nexo de unión entre la computación tradicional basada en números y el mundo real basado en palabras. Por ejemplo, se ha utilizado con éxito para el procesamiento de consultas difusas mediante el empleo de

términos lingüísticos y relaciones de similitud [8] o para la mejora de los sistemas de recuperación de información mediante el uso de recursos lingüísticos (ontologías y/o tesauros) [7].

2. Conceptos preliminares

2.1 Lógica Difusa

Aristóteles introdujo las Leyes del Conocimiento, las que posteriormente serían el sustento de la Lógica clásica. Sus tres leyes fundamentales eran:

a) Principio de la Identidad

b) Ley de Contradicción

c) Ley del Tercero Excluido

Platón dijo que había una tercera región entre verdadero y falso: los grados de pertenencia.

Fue Jan Lukasiewicz el primero que propuso una alternativa sistemática a la lógica bivaluada de Aristóteles, una lógica de vaguedades. La describió como la lógica de los 3 valores, con el tercer valor siendo “Posible”.

Black define en 1937 el primer conjunto difuso mediante una curva que recogía la frecuencia con la que se pasaba de un estado a su opuesto.

En los 60 Lotfi Zadeh, basado en las ideas de Black, creó la ‘Lógica difusa’ que combina los conceptos de lógica y de los conjuntos de Jan Lukasiewicz mediante la definición de grados de pertenencia [1].

La lógica difusa etiqueta las cosas que no pueden ser medidas exactamente, por ejemplo: alto, muy alto, los cuales a estos no se les pueda dar un valor concreto, aquí en donde entra la lógica difusa, donde a estas etiqueta puede

dársele un cierto valor, que se conoce como función de membresía (μ(x)), el cual tiene valores entre 0 y 1 donde pertenecen a un cierto conjunto de características el cual posee un grado ambigüedad, como lo podemos observar en la Figura No. 1. [1]

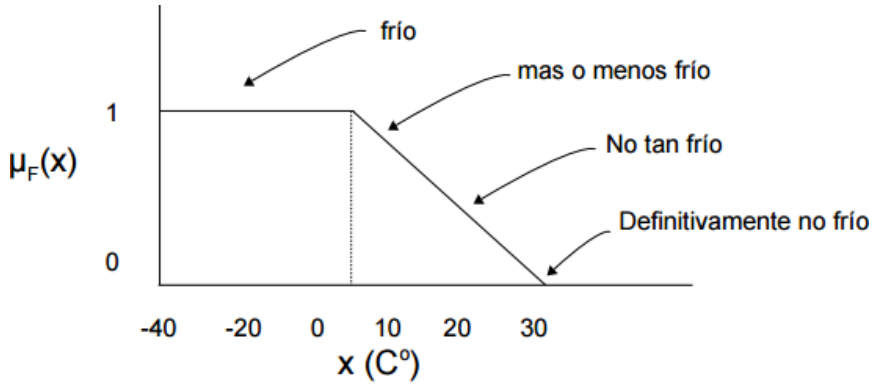

Fig. 1. Grafica de función de membresía.

2.2 Sistema de inferencia de control difuso

Defusificación

Este proceso permite tomar varios valores reales para convertirlos en valores difusos, en donde se asignan grados de pertenencia a cada una de las variables de entrada con relación a los conjuntos difusos previamente definidos utilizando las funciones de pertenencia asociadas a los conjuntos difusos, en la Figura No. 2, vemos la representación [4].

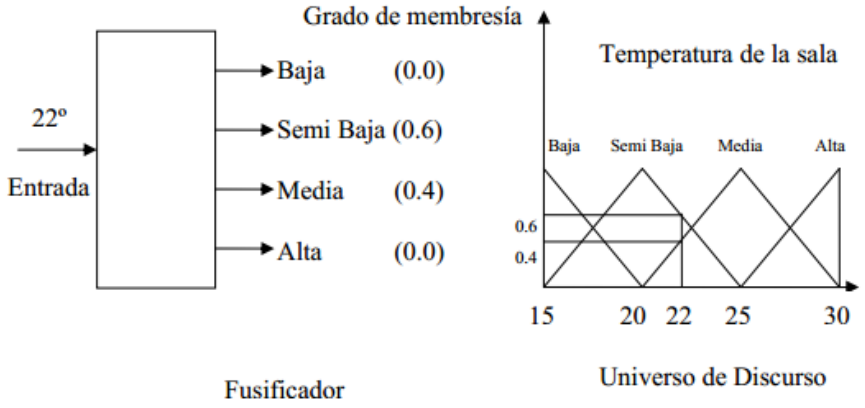

Fig. 2. Entradas a un sistema difuso.

2.3 Datalog

Tomando como referencia a los autores [2,3]. Datalog es un lenguaje lógico desarrollado para el modelo relacional Datalog sin recursión tiene el mismo poder expresivo que el álgebra relacional: sin embargo, a partir de 1999 SQL ha usado una solución para la recursión en Datalog para el desarrollo de consultas recursivas. Es similar a Prolog en su sintaxis, pero su semántica operacional es diferente, una regla o clausula en Datalog tiene la forma:

$$cabeza \leftarrow cuerpo$$

Donde la cabeza es un átomo y cuerpo es una lista de átomos que puede ser vacía; en este caso se habla de un hecho. Los hechos se definen:

$$P(t_1, \ldots., t_n)$$

Donde P es un símbolo de predicado y t_i son variables constantes. No se admiten símbolos de función en t_i, a diferencia de Prolog.

2.4 Reglas lógicas

Una regla se escribe:

$$P \leftarrow Q_1, \ldots, Q_n$$

Dónde: "Si $Q_1, Q_2, \ldots y\ Q_n$ son ciertos, entonces P es cierto". Si $n = 0$ "P es cierto", y se escribe: P.

Para definir el significado de las reglas, existen diferentes alternativas que se mencionan a continuación.

Interpretación de la teoría de pruebas. Es el conjunto de todos los hechos que se pueden probar a partir de las reglas del programa usándolas de todas las formas posibles.

Interpretación de la teoría de modelos. La interpretación de una colección de predicados asigna cierto o falso a cada posible instancia de los predicados, donde los argumentos se escogen de un conjunto infinito de constantes. La interpretación se representa habitualmente por el conjunto de instancias verdaderas.

Definición computacional. La última forma de definir el significado de las reglas lógicas es proporcionar un algoritmo para ejecutarlas para determinar

si un hecho es cierto o falso. Prolog define su semántica de esta forma, solo que tiene un inconveniente, hay hechos que no se pueden probar de esta forma (ramas infinitas).

Datalog es una versión de Prolog, que fue adecuada para dar respuesta a las bases de datos, pero se diferencia en:

1. Datalog no admite símbolos de función en los argumentos.
2. El significado de los programas Datalog sigue el punto de vista de la teoría de modelos, en cambio, Prolog se basa en un significado computacional que se desvía de los significados de la teoría de modelos y la teoría de pruebas.

2.5 El modelo de datos de Datalog es similar al relacional

Una relación se representa por un predicado. Sin embargo, sus argumentos siguen una notación posicional no explicita como el modelo relacional.

Ejemplo 1: Una instancia r de la relación r(A,B) en el modelo relacional definida por r, sería de la siguiente manera:

r

A	B
1	2
3	4

Sin embargo, representada en Datalog por los hechos: r(1,2) y r(3,4).

En Datalog, el primer argumento de R corresponde con el atributo A y el segundo a B.

El significado de la relación en ambos modelos de datos es el mismo, el conjunto de tuplas {(1,2), (3,4)}. Es decir, hay una relación de tipo R entre 1 y 2 y entre 3 y 4.

3. Metodología

Para empezar a construir la interfaz gráfica de base datos deductivas difusas, se hizo uso de los lenguajes de programación de Swi-prolog [9], MYSQL[6] y Netbeans[5] software de GNU licencia libre, para utilizar el motor de inferencias y Netbeans para la interfaz de usuario, se realizó primero la conexión de Swi-Prolog, como lo podemos observar en la siguiente Figura No. 3.

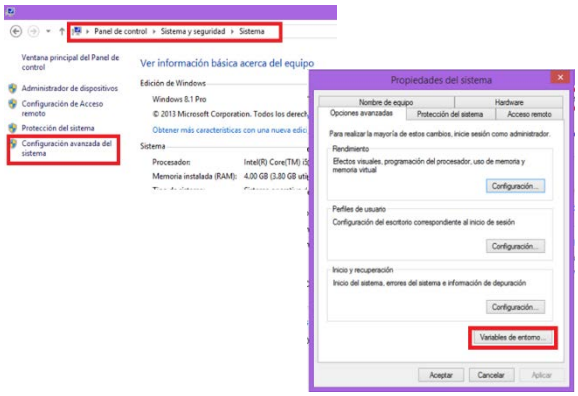

Fig. 3.Conexiones de software.

En la conexión se deben de considerar algunos requisitos, antes de comenzar, como tener instalado Swi-prolog y Netbeans, revisando el path de sistema, se realizó la configuración necesaria para su uso correcto, se hizo necesario realizar estas configuraciones ya que son la que nos permitirán interactuar con ambos lenguajes.

3.1 Conexión MYSQL con SWI-PROLOG

El primer paso es crear una base de datos para realizar la conexión entre MySQL y Swi-Prolog, en este caso se realizo una base de datos llamada Escuela, y el objetivo principal proporcionará el promedio de cada alumno. Para probar las conexiones se creo un demo con varias tablas, se pudo haber creado una base de datos con una sola tabla para mayor simplicidad, ya que lo importante es ver como se realiza la conexión. Una vez que se tiene creada la tabla, el siguiente paso que fue conseguir el origen de datos a través de ODBC para MySQL. Como lo podemos observar en la Figura No. 4.

Fig. 4. Creación de la Base de Datos.

Una vez llenado este formulario, iremos a prolog, donde teclearemos el código para abrir, y cerrar la conexión con MySQL. Se tuvieron que realizar los predicados correspondientes para la conexión que se muestra en la Figura No. 5.

conexion.pl

```
%************CONEXION DE PROLOG A MYSQL***********
abrir_conexion:-
        odbc_connect('escuela_promedio',_,
                                [user(root),
                                password(''),
                                alias(escuelapromedio),
                                open(once)
                                ]).

cerrar_conexion:-
        odbc_disconnect('escuelapromedio').
```

Fig. 5 clausulas de prolog, para la conexión.

En la primer línea va el nombre de la fuente de datos, y en alias puede ir el nombre que se desee, siempre y cuando no tenga caracteres especiales ni espacios. Una vez estableciendo esto, se compilo y se ejecutó el código para corroborar que la conexión haya resultado un éxito como se ve en la Figura No. 6.

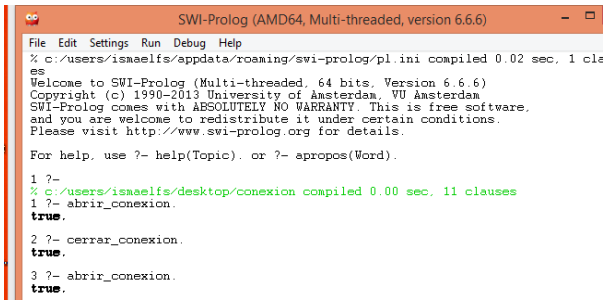

Fig. 6 Conexión desde Prolog a la Base de datos.

Para trabajar las bases de datos deductivas difusas, se realizaron una serie de predicados dinámicos, con la finalidad de que se pudieran leer y lanzar el grado de pertenencia de las consultas en la base de datos.

:- op(1115, fx, if),

:- op(1150, xfx, (else)) ,

:- op(1110, xfy, then),

:- op(1100, xfy, or),

:- op(1000, xfy, and),

:- op(900, fx, not),

:-op(95,xfx, :&),

:-op(700, xfx, (is))

Para la construcción de los conjuntos difusos, se realizaron predicados dinámicos, que nos ayuden a modelizar la base de datos.

Para el dominio del discurso, se implementó un predicado $set\_universe/4$, el cual actúa de manera dinámica.

$$set\_universe(Domain, [\, Range1, Range2], Measure)$$

Donde $Domain$ es el dominio del discurso, $Range1$ el límite inferior y $Range2$ el límite superior, donde $Range1 < Range2$, la $Measure$ es el parámetro de medición.

Para la implementación de los subconjuntos difusos, se utiliza el predicado $subset\_fuzzy$, el cual está definido de forma dinámica:

$$subsets\_fuzzy(\, VL, [R1, R1, R3])$$

Donde VL es el nombre de la variable del conjunto difuso, y $R1, R2$ y $R3$, son los valores que se utilizan en los subconjuntos difusos, cumpliendo los valores con la siguiente restricción: $R1 < R2 < R4$, de acuerdo a la función de pertenencia que se puede definir: trapezoidal o triangular. La función de pertenencia de forma triangular la da por default y en el caso de la función trapezoidal la restricción a cumplir es $R1 < R2 <= R3 < R4$.

Una vez realizada la introducción de las variables del discurso de universo μ, los subconjuntos difusos son definidos internamente como una función de pertenencia que enlaza o empareja los elementos de un dominio o universo del discurso X con valores de verdad en el intervalo [0,1], con el predicado $leeValue/1$ y con el predicado $cons\_sets\_fuzzy/6$.

$$cons\_sets\_fuzzy(domain, VL([(R1, V), (R2, V), (R3, V)])$$

Donde $domain$, es el universo de discurso, VL nombre del conjunto difuso y las variables $R1, R2, R3$ son los elementos del conjunto difuso y V el valor de verdad.

4. Resultados

Para el proyecto se realizó una interfaz en Java, la cual fue probada con una base de datos denominada Escuela, donde se trabajó en los promedios de los alumnos, en la Figura No.7, se muestra una ventana donde podemos realizar una conexión a la base de datos, registrar al alumno, ver su promedio y registrar la materia.

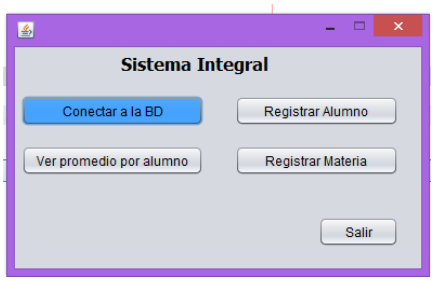

Fig. 7 Acceso principal a la interfaz.

Una vez realizada la conexión podemos acceder en agregar materia y nos muestra la ventana de la Figura No. 8, registro de materias, con la opción de consulta.

Fig. 8. Agregación de datos a la B.D.

La Figura No. 9, permite ver el promedio, por lo que cuenta con una caja de texto para introducir cualquier número de control de alumnos, y nos arrojará su promedio, si y sólo si el alumno está dentro de la base de datos.

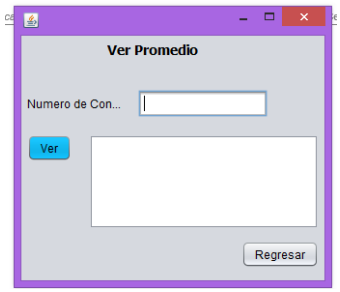

Fig. 9. Consulta de Promedio.

La Figura No. 10, nos muestra como podemos hacer el llamado de los predicados para tratar la base datos deductiva difusa, a través de Prolog y MySQL.

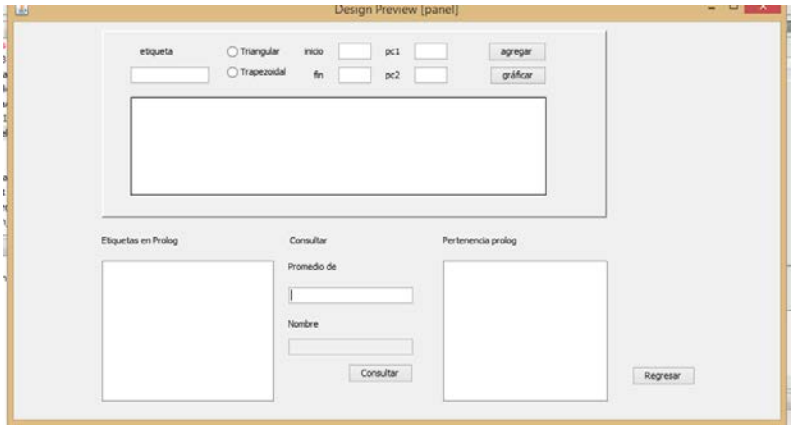

Fig. 10. Interfaz para el control de conjuntos difusos de la B.D.

5. Conclusión y trabajo futuro

La enorme cantidad de información que se maneja hoy en día nos lleva a buscar nuevas opciones para su manejo, pero sobre todo cuando además de ser grandes bases de datos vienen con incertidumbre, provoca a los

investigadores de esta materia seguir realizando nuevo software que ayude a modelizar, optimizar esas bases de datos que nos arrojen información con una precisión más exacta. Al realizar esta interfaz se cumplen las expectativas de darle al lector herramientas que pueden ayudar a manejar base de datos deductivas difusas y puedan trabajar con ellas de una manera confiable y amigable, pues muchas veces el no contar con una implementación de alto nivel, desalientan a los usuarios a la explotación real de las bases de datos por no contar con una interfaz que sea de fácil manejo. Esta es una primera parte de la propuesta, considerando que es un primer acercamiento para trabajar con bases de datos que además de ser deductivas sean difusas, con la finalidad de tener un mejor manejo de la información de cualquier índole.

Bibliografía

[1] Curso Introductorio de Conjuntos y Sistemas Difusos (Lógica Borrosa y Aplicaciones), por el Dr. José Galindo G., Universidad de Málaga (España)

[2] Date, C.J. Introducción a los sistemas de bases de datos, Person, Prentice hall, 2001.

[3] Gallaire, H. and Minker J., editors. Logic and Data Bases, Advances in Data Base Theory. Plemum Press, 1978.

[4] Kandel Abraham (1991) Fuzzy Expert systems. CRC Press, Tampa Florida.

[5] https://netbeans.org/

[6] https://www.mysql.com/

[7] P.J. Garcés, J.A. Olivas and F.P. Romero. FISS: Aplicación de la Relación Borrosa de Sinonimia y de Ontologías Borrosas a un Meta buscador. Congreso Español de Tecnologías y Lógica Fuzzy (2002).

[8] S.M. Chen and H.R. Hsiao. A New Approach for FuzzyQuery Processing Based on Automatic Clustering Techni-ques. Information and Management Sciences, 18(3):223–240 (2007).

[9] SWI-Prolog. Disponible en http://www.swi-prolog.org.