ARTÍCULO ORIGINAL
GESTIÓN DEL CONOCIMIENTO

Búsqueda de predicados difusos en una base de datos utilizando metaheurísticas

Search of fuzzy predicates in databases using metaheuristics

Taymi Ceruto-Cordovés[I], Alejandro Rosete-Suárez[I], Rafael Espín-Andrade[II]

[I] Instituto Superior Politécnico José Antonio Echeverría, Cujae. La Habana, Cuba.
E-mail: tceruto@ceis.cujae.edu.cu, rosete@ceis.cujae.edu.cu
[II] Universidad de Occidente, Sinaloa, México.
E-mail: rafaelespin@yahoo.com

Recibido: 05/09/2011
Aprobado: 03/10/2013

RESUMEN
Actualmente se ha incrementado de forma paralela tanto la cantidad de información almacenada como la necesidad de desarrollar algoritmos que permitan extraer conocimiento útil de la misma de forma automática. Estos algoritmos se incluyen dentro del área de extracción de conocimiento en bases de datos y están enfocados en un modelo determinado (reglas, grupos, árboles). El desarrollo de modelos adecuados puede conducir a la obtención de decisiones exitosas y es por ello que en este trabajo se propone un enfoque más flexible para representar el conocimiento extraído de una base de datos. El método presentado se fundamenta en el paradigma de la lógica de predicados y el cálculo de sus valores de verdad mediante operadores de lógica difusa. Además incluye un proceso de optimización que permite obtener y mejorar los resultados obtenidos.

Palabras claves: descubrimiento de conocimiento en bases de datos, lógica difusa, metaheurísticas.

ABSTRACT
Nowadays, both the quantity of information stored and the necessity of developing algorithms in order to extract useful knowledge of this information automatically have been increased in a parallel way. These algorithms are included inside the area of extraction of knowledge in databases and they are focused on a certain model (rules, groups, trees). The development of appropriate models can lead to obtaining successful decisions and for that reason this paper proposes a more flexible way to represent the extracted knowledge of databases. The presented method is based on the paradigm of predicates logic and the calculation of its values using fuzzy logic operators. It also includes a process of optimization that allows to obtain the results and to improve them.

Key words: *knowledge discovery in databases, fuzzy logic, metaheuristics*.

I. INTRODUCCIÓN

En la actual sociedad de la información, donde cada día se multiplica la cantidad de datos almacenados de forma exponencial, la Minería de Datos (*Data Mining*, DM) es una herramienta fundamental para analizarlos y explotarlos de forma eficaz. Este es un campo multidisciplinar que cubre numerosas áreas y se aborda desde múltiples puntos de vista; y está enfocado a encontrar tendencias relevantes, modelos y relaciones entre aquellos aspectos que no son visibles [1].
En DM existen diferentes modelos para representar el conocimiento extraído (árboles de decisión, reglas, grupos) y a cada uno de ellos se puede llegar aplicando diferentes algoritmos (*ID3*, *Apriori*, *K-Medias*, entre otros) de acuerdo al tipo de tarea (clasificación, agrupamiento, asociación, entre otros) que se esté realizando. Ninguno de estos modelos brinda la posibilidad de obtener conocimiento donde por ejemplo la relación lógica principal sea la doble equivalencia o la conjunción o la disyunción o cualquier combinación libre de operadores entre variables de una base de datos. Es por ello que en este trabajo se propone un nuevo modelo basado en predicados difusos para la representación del conocimiento extraído de una base de datos. Este enfoque es más flexible y brinda la posibilidad de descubrir conocimiento que ninguno de los métodos anteriores permite, por estar enfocados en un modelo determinado y tener restricciones asociados a los mismos.

II. MÉTODOS

El proceso de extracción de conocimiento en bases de datos ha ido evolucionando con el paso del tiempo y continúa haciéndolo a través de la investigación interdisciplinar de diferentes áreas. De forma particular, las metaheurísticas y la lógica difusa (*Fuzzy Logic*, FL) no solo tienen la posibilidad de resolver problemas propios de la minería de datos, sino también problemas de otras etapas del proceso de Descubrimiento de Conocimiento en Bases de Datos (*Knowledge Discovery in DataBases*, *KDD*) [1].
Estas metodologías se han visto involucradas fundamentalmente en el descubrimiento de reglas (Si-Entonces) debido a su simplicidad, comprensibilidad y capacidad de expresión. Esta aproximación es el antecedente más cercano conocido (aunque difieren) a la propuesta que se realiza en este trabajo. Ahora, dependiendo del conocimiento que se descubre, se pueden distinguir diferentes tipos de reglas como son: reglas de clasificación, reglas de asociación, etc. Específicamente en el descubrimiento de reglas los algoritmos más utilizados han sido los algoritmos genéticos (*Genetic Algorithms*, GA) y la programación genética (*Genetic Programming*, GP) [1].
En los GA los individuos se representan como una cadena lineal, y en el caso de reglas cada eslabón suele ser una pareja atributo-valor, mientras que en GP un individuo suele representarse mediante un árbol, y en este caso particular los nodos hoja o terminales son condiciones de reglas y/o valores de atributos, y los nodos internos representan las conectivas.
En el contexto de descubrimiento de reglas utilizando algoritmos evolutivos, un individuo corresponderá a una regla o conjunto de reglas candidatas (dependiendo del enfoque utilizado); la función de evaluación corresponderá a alguna medida de la calidad de la regla o conjunto de reglas; el procedimiento de selección utilizará la evaluación dada a cada individuo para seleccionar las mejores reglas o conjunto de reglas; los operadores genéticos transformarán una regla candidata en otra.
Los algoritmos evolutivos realizarán una búsqueda en el espacio de reglas candidatas como haría un método de inducción de reglas. La principal diferencia entre los algoritmos evolutivos empleados para el descubrimiento de conocimiento y los algoritmos de inducción de reglas es la estrategia de búsqueda empleada. En efecto, los algoritmos clásicos de aprendizaje inductivo suelen utilizar una estrategia voraz de búsqueda local, mientras que los algoritmos evolutivos utilizan una estrategia de búsqueda global inspirada en la selección natural.
Cuando los atributos de una regla son del tipo numérico como por ejemplo la estatura, el sueldo, o categóricos como el color; tienen pocos valores (sus correspondientes dominios son finitos y reducidos). En este caso la transformación es directa y poco problemática, pero cuando son muy numerosos o cuando estos atributos son de tipo numérico continuo (real, racional), no es posible especificar en forma sencilla jerarquías para disminuir el nivel de granularidad, en este caso se habla de trabajar con atributos de tipo cuantitativo y a las reglas de asociación que involucran

ítem sobre estos atributos se les denomina reglas de asociación cuantitativa (*Quantitative Association Rule*, QAR) [2].
Las reglas de asociación cuantitativas se basan en dividir el dominio del atributo en intervalos y después extraer las reglas cuyos *ítems* son los pares <atributo;intervalo> en lugar de <atributo;valor>. Las reglas de asociación cuantitativas han sido desarrolladas por un conjunto amplio de autores y las mismas también han estado vinculadas al uso de metaheurísticas como: AG, ACO [3; 4; 5; 6].
Pero al dividir el dominio en un número de intervalos surge el llamado problema de la frontera en el que se pueden rechazar algunos intervalos interesantes al excluir algunos elementos potenciales cerca de su frontera. Para solucionar este problema algunos autores han propuesto dividir el dominio del atributo en intervalos solapados [2].
Justamente con la lógica difusa se puede evitar el problema de los bordes del intervalo utilizando conjuntos difusos en lugar de intervalos [7]. De esta forma un elemento pertenecería a un conjunto difuso con un grado de pertenencia en el intervalo [0;1]. Además se puede dotar a dichos conjuntos de una semántica significativa para el entendimiento del usuario utilizando para ello términos lingüísticos (<atributo;etiqueta>).
De ahí que surgieran las reglas de asociación difusas a partir de datos cuantitativos (*Mining Quantitative Association Rules using Fuzzy Set Theory*) donde en la mayoría de los casos se asume que las funciones de pertenencia (FP) se conocen previamente. Debido a la importancia de la definición de las funciones de pertenencia y su impacto en los resultados, otro conjunto de investigadores han realizado también un aprendizaje o ajuste de las FP [8].
Dada la panorámica descrita anteriormente, se hace imprescindible, por un lado, un análisis exploratorio de los datos y por otro lado, el empleo de métodos robustos y flexibles. Es por ello que el algoritmo que se propone en este trabajo tiene como característica diferenciadora que el aprendizaje puede ser no supervisado, los predicados pueden cambiar su forma representando distinto conocimiento, se pueden elegir diferentes operadores difusos tratando de que sean lo más apropiado posible para la toma de decisiones y por último y no menos importante, los algoritmos utilizados para la búsqueda no se limitan al uso exclusivo de los tan populares algoritmos evolutivos. Todos estos aspectos serán detallados a continuación utilizando tres epígrafes: esquema de representación, función de evaluación y proceso de búsqueda. Luego se muestra la experimentación realizada y las conclusiones obtenidas.

1. Esquema de representación.

La elección del esquema de representación depende, entre otros aspectos del tipo de conocimiento a descubrir. En este caso se utilizarán predicados, como una estructura flexible que facilita la extracción de conocimiento general.
El alfabeto proposicional consta de los siguientes aspectos [9]:

1. Un conjunto de variables denominadas átomos.
2. Un conjunto de conectivos lógicos:
   ¬ (negación): invierte el sentido de la proposición.
   ∨ (disyunción): cualquiera de las proposiciones es cierta.
   ∧ (conjunción): las proposiciones son ciertas simultáneamente.
   → (implicación = condicional = si…entonces): el cumplimiento o la verdad de una de las proposiciones tiene como consecuencia el cumplimiento de la otra.
   ↔ (equivalencia = bicondicional = si y sólo si)
3. Los símbolos de paréntesis como símbolos auxiliares, para evitar ambigüedades, aunque se puede prescindir de ellos con un convenio de precedencia.

También pueden incluir un conjunto de modificadores (*hedges*): "muy", "algo". Estos operadores modifican el valor de verdad de una proposición intensificando, moderando y ejerciendo otros efectos. Los modificadores más utilizados son funciones de la forma $f(x)=x^a$ donde a es un exponente mayor o igual que cero. Suelen utilizarse por ejemplo los exponentes 2 y 3 para modelar las palabras "muy" e "hiper", y el exponente ½ para modelar los conceptos "algo". Cuando se utiliza un valor de a>1 se dice que se realiza una concentración, cuando es menor que 1 es una dilatación.
Los modificadores "muy" e "hiper" son más exigentes y por tanto el valor de verdad no aumenta hasta que no esté realmente cercano a uno. Por otro lado el modificador algo favorece el valor de verdad en todo momento. Una de las cosas a tener en cuenta en los resultados del método que se

propone es la tendencia a obtener soluciones con el modificador algo, lo cual habrá que convertir en una restricción si fuera necesario.

Una vez definido el fenotipo (solución decodificada que representa una solución en el contexto del problema) es necesario definir el genotipo (solución codificada). En la bibliografía se pueden encontrar múltiples formas de lograr una representación codificada de una solución. En particular cuando se trata de descubrir reglas se hace alusión a los siguientes enfoques [10]:

- Enfoque Pittsburgh: un conjunto de reglas se codifica en un único individuo. Este tiene como inconveniente que el individuo es muy grande y se hace costoso computacionalmente.
- Enfoque Michigan: un cromosoma codifica una única regla, pero la solución final es una población; por lo que es necesario evaluar el comportamiento del conjunto de reglas al completo y la aportación de la regla individual. Tiene como ventaja que la sintaxis es más reducida, pero resulta más complicado evaluar la calidad del conjunto de reglas como un todo.
- Enfoque IRL (*Iterative Rule Learning*): cada cromosoma representa una regla y la solución global está formada por los mejores individuos de una serie de ejecuciones sucesivas.

En este trabajo se propone utilizar algo similar al enfoque IRL; un cromosoma codifica un único predicado y la solución global será el conjunto de los mejores predicados obtenidos en ejecuciones del algoritmo. No habría que diferenciar entre antecedente y consecuente, pero si es necesario definir si el cromosoma será de longitud fija o variable. En este caso se empleará un cromosoma de longitud fija que utiliza codificación entera. Aunque en realidad el tamaño del predicado es variable, ya que al conjunto de valores válidos de cualquier variable se le añade un valor especial que indica la ausencia de dicha variable en el predicado.

Se realiza la codificación de un predicado, el cual se muestra en la tabla 1:

Tabla 1. Codificación de un predicado.

Escala	Genotipo	Fenotipo
0 – no está en el predicado 1 – está en el predicado 2 – aparece negada 3 – aparece con el modificador muy	X_0 | X_1 | X_2 | X_3 | X_4 1 | 0 | 2 | 3 | 1	$X_0 \wedge \neg X_2 \wedge (X_3)^2$

En el ejemplo anterior se utiliza una escala que determina el dominio de las variables. El genotipo ejemplo se decodificaría de la siguiente manera para llegar al fenotipo correspondiente:

- la variable X_0 aparece normal.
- la variable X_1 no interviene (está marcada con el valor cero).
- la variable X_2 aparece negada.
- la variable X_3 aparece afectada por el modificador muy (implica elevar al cuadrado).

El predicado obtenido representa una conjunción de proposiciones individuales, aunque también se pueden representar predicados en disyunción. En este esquema de codificación se logra utilizando un gen adicional (último representado en este caso por la variable ficticia X_4) que determina el tipo de relación entre las variables. De esta forma el algoritmo es más flexible, ya que este gen también puede evolucionar. Además es un parámetro menos a determinar por el usuario, que traería como consecuencia que habría una mayor cantidad de combinaciones (crece el espacio de búsqueda).

Por otro lado, en el ejemplo mostrado, ninguna variable se repite, debido a que se empleó una sola cláusula. En este trabajo se ha optado por una representación de variables basada en formas normales (conjuntiva o disyuntiva). De esta forma las variables podrían repetirse, explicitando que se utilizarán 2 o más cláusulas.

En la literatura consultada por los autores, una aproximación cercana es el descubrimiento de reglas en forma normal disyuntiva, reglas DNF (conjunción de atributos y disyunción entre los valores de los atributos) [8]. En este tipo de representación se utiliza un esquema de codificación binaria con tantos genes por variable como valores posibles existían para la misma. La idea de utilizar disyunción de valores para una variable, les permitió a los autores extraer reglas más genéricas en un contexto donde las variables eran discretas y el número de valores perdidos era muy elevado.

Cuando se utilizan formas normales sólo hay tres tipos de conectivas lógicas [11]:

- Conjunción.
- Disyunción.
- Negación (sólo afectando a los átomos).

Si la conectiva principal es la conjunción, entonces es una forma normal conjuntiva (*Conjunctive Normal Form*, CNF). Si por el caso contrario, la conectiva principal es la disyunción, entonces es una forma normal disyuntiva (*Disjunctive Normal Form*, DNF), como el ejemplo que se muestra a continuación.

Tabla 2. Ejemplo de predicado en DNF.

Genotipo									Fenotipo
X_0	X_1	X_2	X_3	X_0	X_1	X_2	X_3	X_4	$(X_2 \wedge \neg X_3) \vee (X_0 \wedge \neg X_1)$
0	0	1	2	1	2	0	0	1	

Para cualquier fórmula en lógica clásica se puede encontrar una fórmula equivalente en CNF o DNF. Esto es posible mediante el uso de un conjunto de leyes y propiedades como: Leyes de De Morgan y las que permiten la supresión de implicación y la doble implicación.
Esta no es la única forma de representación, también se pueden utilizar árboles. Por ejemplo un predicado se puede representar utilizando un árbol general (para evitar asociatividad) donde cada nodo puede ser un operador ($\wedge$,$\vee$,$\rightarrow$,$\leftrightarrow$) ó un átomo (p, q). Esta variante está siendo implementada por Prolog debido a su carácter recursivo y potencialmente descriptivo [12].

2. Función de evaluación.

En el proceso de descubrimiento de conocimiento se intentan conseguir predicados con alto valor de verdad. Este valor depende estrechamente del conjunto de datos donde el predicado es evaluado. En nuestra propuesta, la función de adaptación se calcula utilizando operadores de lógica difusa, debido a que todas las variables presentes en el problema son tratadas como variables lingüísticas que pueden tomar un valor cualquiera de veracidad dentro de un conjunto de valores que oscilan entre dos extremos, la verdad absoluta (1) y la falsedad total (0); que luego son traducidos a etiquetas lingüísticas.
Una característica a destacar en la lógica difusa es que no existe una definición única de algunas de las operaciones clásicas como la unión (disyunción) o la intersección (conjunción) de conjuntos, sino que existen múltiples formas de desarrollar estas operaciones. En la intersección se utiliza una norma triangular (T-norma). Algunos de los operadores asociados frecuentemente a la intersección son el Mínimo y el Producto, aunque existen otros. Algo similar sucede con la unión, en la cual se utiliza una conorma triangular (T-conorma o S-norma). Aquí los operadores asociados con mayor frecuencia son el Máximo y la Suma Algebraica, aunque también existen muchos más. Para el caso del complemento (negación) la función es una C-norma, y aunque tiene igualmente varias funciones asociadas (propuestas por Sugeno, Yager, etc), existe un mayor consenso para el uso de $n(x) = 1 - x$ [13; 14].
Los operadores que se utilizan con mayor frecuencia tanto en las normas, como en las conormas, tienen como limitaciones la asociatividad (operadores conjunción y disyunción). Por ejemplo en un sistema asociativo los siguientes predicados:
$(p \wedge q) \wedge r = p \wedge (q \wedge r) = p \wedge q \wedge r$; representarían lo mismo, cuando en realidad no lo son. Como hay preferencias, representan árboles de decisión diferentes y la veracidad de dichos predicados no debe dar el mismo resultado.
Si un sistema es asociativo o no es sensible o no es idempotente. Por eso es importante que los sistemas no sean asociativos, ya que dificultan el uso de los enfoques lógicos en la modelación de la decisión [15].
Por ejemplo el par de operadores Min-Max es idempotente ($p \wedge p = p$; $p \vee p = p$), pero representa un modelo no sensible. Importantes cambios en los valores de verdad de los predicados básicos no modifican el valor de verdad del predicado compuesto. Por ejemplo el valor de verdad de los predicados que se muestran a continuación es el mismo: $0.5 \wedge 0.5 = 0.5$; $0.5 \wedge 0.8 = 0.5$, cuando en realidad el valor de la segunda variable es diferente, lo cual no parece justo.

El par de operadores Producto–Suma Algebraica si representa un modelo sensible ($0.5 \wedge 0.5 = 0.25$; $0.5 \wedge 0.8 = 0.4$), pero no es idempotente. Esta característica trae problemas a la hora de interpretar los valores de verdad obtenidos. En el primer ejemplo no parece justo obtener un valor de verdad más falso que verdadero (0.25) por debajo del valor real de las variables, cuando en realidad ahí la vaguedad es la máxima.

Además todos estos operadores (comúnmente más utilizados) tienen una falta total de compensación de los valores de verdad de los predicados básicos cuando se calcula la veracidad de los predicados compuestos. En el caso de Min-Max el valor de verdad de una conjunción siempre recae en el valor mínimo de cualquiera de sus variables, independientemente de que el resto tenga un alto valor. En el caso de Producto–Suma Algebraica el valor de verdad de una conjunción siempre es menor que el valor mínimo de cualquiera de sus variables.

Es por ello que se consideró necesario el estudio de operadores compensatorios y en particular de la Lógica Difusa Compensatoria (*Compensatory Fuzzy Logic*, CFL). La misma es sensible ante cambios en cualquiera de las variables, es idempotente, permite la compensación de los valores de unas variables con otras y facilita la interpretación de los resultados. Esto es posible debido a que en esta lógica se renuncia a la asociatividad. Esta última característica hace que los operadores de conjunción (Media Geométrica) y disyunción (Dual de la Media Geométrica) no cumplan las condiciones para ser T-Norma, ni S-Norma [15].

Para el cálculo de la veracidad de un predicado también se utilizan cuantificadores. Estos se usan para medir (o cuantificar) la cantidad o la proporción de objetos o elementos que cumplen o satisfacen cierta condición. En lógica clásica existen dos muy importantes [9]:

- Universal (Para todo): Se refiere a todos los elementos u objetos.
- Existencial (Existe): Se refiere al menos a uno de los elementos u objetos.

Estos cuantificadores deben ser introducidos de manera natural a partir de los operadores conjunción y disyunción. En esta tesis en particular, se hace uso del cuantificador universal, para lograr que el predicado obtenido sea válido para todas las tuplas o registros de la vista minable. Esto tiene como inconveniente que los resultados obtenidos son muy exigentes, ya que en este caso lo que se realiza es una conjunción.

El objetivo global de la función de evaluación es orientar la búsqueda hacia predicados que maximicen el valor de verdad, y este se calcula siguiendo el procedimiento que se muestra en la figura 1.

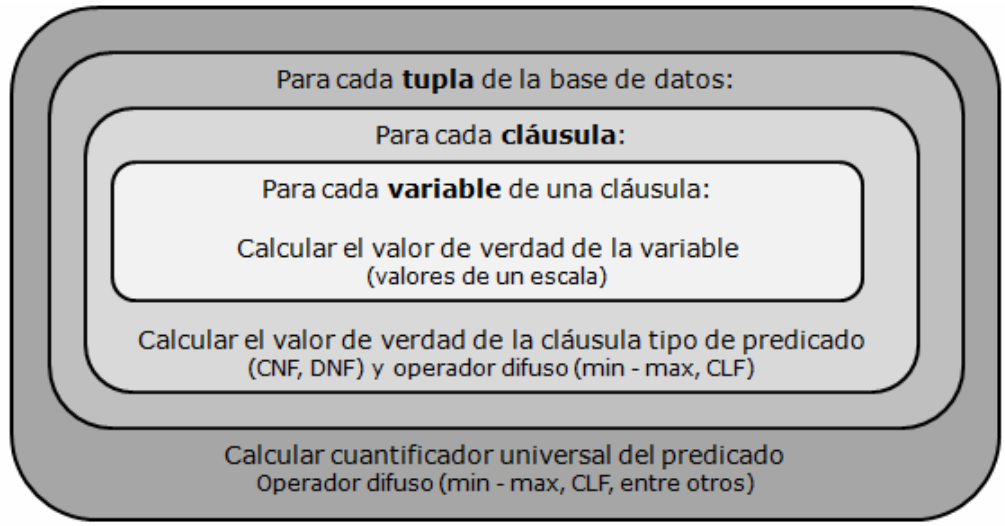

Figura 1. Cálculo del valor de verdad de un predicado.

La complejidad del algoritmo en su peor caso es polinomial O (n * m * p) donde:

- n: cantidad de tuplas o registros.
- m: cantidad de cláusulas.
- p: cantidad de variables de una cláusula.

Ahora, este algoritmo polinómico sólo permite comprobar si una posible solución es válida o no. El problema en sí intenta descubrir predicados difusos y por sus características necesita un método

de resolución no determinista (metaheurísticas) para obtener soluciones hipotéticas que se van desestimando (o aceptando) a ritmo polinómico. Por lo que se puede afirmar que el problema a resolver es NP (la N de no-deterministas y la P de polinómicos) [16].

3. Proceso de búsqueda de los predicados.

Para el proceso de búsqueda de los predicados se decidió utilizar metaheurísticas. En primer lugar porque no existe un algoritmo exacto que lo haga y segundo porque el espacio de soluciones puede llegar a ser grande, teniendo en cuenta los aspectos que determinan la complejidad del problema:

- la cantidad de variables involucradas, que a su vez depende de la cantidad de cláusulas del predicado.
- los posibles valores que puede tomar cada variable.

Independientemente de que la cantidad de combinaciones no fuera tan grande, de todos modos no es factible desde el punto de vista computacional hacer el cálculo de la función objetivo para todas las combinaciones posibles, por la dimensión que pueda tener la base de datos. Por ejemplo, si el problema tuviera: 8 variables, 2 cláusulas, y cada variable pudiera tomar 5 valores; el tamaño del espacio de búsqueda sería: 5^{16} = 152587890625.

Hay diferentes formas de clasificar y describir las técnicas metaheurísticas: inspiradas en la naturaleza o no, deterministas o estocásticas, iterativas o greedy, entre otras. En este trabajo serán clasificadas de acuerdo a si en cada paso manipulan un único punto del espacio de búsqueda o si operan sobre un conjunto (población) de ellos. Entre los basados en un punto (o en trayectoria) se encuentran: búsqueda aleatoria (*Random Search*, RS), GRASP (*Greedy Randomized Adaptive Search Procedure*) y los escaladores de colinas (*Hill Climbing*, HC), búsqueda tabú (*Tabu Search*, TS), recocido simulado (*Simulated Annealing*, SA), entre otros. Algunos de los basados en población son: estrategias evolutivas (*Evolution Strategies*, EE), algoritmos genéticos (*Genetic Algorithms*, GA), algoritmos de estimación de distribuciones (*Estimation of Distribution Algorithms*, EDA), entre otros [17; 18; 19; 20].

Como se puede apreciar hay muchos métodos metaheurísticos que se pueden usar para resolver problemas combinatorios. Algunos son variantes de otros, introduciendo nuevas ideas o heurísticas. Hay algunos más simples, otros tienen más parámetros. Algunos tienen buena fama y otros están subvalorados. De forma general hay mucha propaganda y sectarismo en este tema; pero la realidad dice que no hay superioridad posible de ninguno sobre otro en general (Teorema *No Free Lunch*) [21].

Según el Teorema NFL "*No Free Lunch Theorem*" ningún algoritmo es mejor que otro en la totalidad de los problemas en los que son aplicables. Es decir, si un algoritmo A es mejor que uno B sobre un grupo determinado de problemas, entonces debe esperarse que existan un conjunto igual de problemas donde ocurra lo contrario [16]. Lo que hace que un algoritmo se comporte mejor o no con respecto a un problema determinado, es la correcta selección de la función objetivo, así como los operadores que permiten variar de un estado a otro del problema. Desgraciadamente este Teorema nadie lo niega, pero todos lo ignoran.

En la Facultad de Ingeniería Informática de la Cujae, específicamente en el Grupo de Inteligencia Artificial (GRIAL) no se considera preponderante ningún algoritmo en particular, sino que prefiere considerarlos todos para determinar cuál es la mejor metaheurística para un problema específico. Esto es posible gracias al desarrollo e implementación de una biblioteca de clases (denominada BiCIAM) que es *Open Source* y que está basada en el paradigma general de solución de problemas en Inteligencia Artificial llamado "generación y prueba" [22]. Para implementar la biblioteca se utilizó como lenguaje de programación *Java*, como herramienta *CASE* para la modelación y generación de código *MagicDraw*, y como entorno integrado de desarrollo *Eclipse*.

BICIAM implementa un modelo integrado que engloba a los algoritmos metaheurísticos, basados en un punto con los poblacionales; de forma tal que se puedan combinar sus características colaborativamente. Este modelo, permite que la búsqueda pueda seguirse como lo haría cada uno de los algoritmos individuales que lo integran, sólo ajustando los parámetros de manera adecuada. Adicionalmente, pueden crearse combinaciones de la ideas de cada uno de los algoritmos para la solución de un problema dado. Además permite comparar las variantes puras. Tiene también como ventaja que separa el problema del algoritmo, lo que hace que el problema (variables, restricciones, operadores, etc.) se codifiquen una única vez y pueda ser utilizado en cualquiera de los algoritmos [22].

El método de extracción de conocimiento que se propone tiene como objetivo general obtener predicados de lógica difusa que expresen información sobre la mayoría de los ejemplos del conjunto de datos. Para ello se hace un esquema iterativo que permite la obtención de varios predicados, mientras no se cumpla un número máximo de iteraciones.
El esquema completo del algoritmo es el siguiente:

```
INICIO
  Cto_Predicados  Ø
  Inicializar parámetros
  P_ACT =  Generar solución inicial
  Cto_Predicados  = Cto_Predicados  + P_ACT
  REPETIR
    P_CAND =  Generar (P_ACT , operador)
    Si se acepta P_CAND
       P_ACT   =   P_CAND
       Cto_Predicados  = Cto_Predicados  + P_CAND
  MIENTRAS Cant_Iteraciones > 0
  Devolver Cto_Predicados
FIN
```

Los parámetros que deben ser inicializados son los siguientes: la base de datos de la cual se extraerá el conocimiento, la cantidad de variables, el par de operadores de lógica difusa a utilizar para las operaciones clásicas, la cantidad de cláusulas, el algoritmo metaheurístico a emplear y el número máximo de iteraciones.
La solución inicial (P_{ACT}) puede ser generada aleatoriamente, ser introducida como parámetro o ser construida con un procedimiento ávido-aleatorio. La misma debe cumplir con el dominio y las restricciones del problema.
La función Generar tiene la responsabilidad de encontrar un estado candidato (P_{CAND}) a partir del estado actual (P_{ACT}) aplicando un operador. Como resultado de aplicar este operador al estado actual varias veces se obtiene una vecindad, en la que los estados generados tienen que cumplir con el dominio y las restricciones del problema. La forma de generar la solución es específica de cada algoritmo metaheurístico y depende en este caso del que haya sido seleccionado como parámetro.
Si el estado candidato es aceptado (depende igualmente de la estrategia del algoritmo), el estado referencia del generador pasaría a ser dicho estado candidato.

III. RESULTADOS

Para comprobar las características de la propuesta, se realizaron experimentos con una pequeña base de ejemplos del mundo real referentes a la economía mexicana. La elección de esta base de datos estuvo condicionada a que la misma ya había pasado por una fase de selección, limpieza y transformación, requerida como precondición de nuestro algoritmo. El proceso estuvo encaminado a que el algoritmo descubriera relaciones en corto tiempo entre las siguientes variables: Inflación (*Inflation, I*), Producto Interno Bruto (*Gross Internal Product*, GIP) y la Paridad (*Parity*, P).
La vista minable consta de 21 tuplas, por lo que se considera un problema relativamente pequeño. No obstante, el tamaño del espacio de búsqueda fue: $9^6 = 531441$ el cual no se considera del todo despreciable. Para calcular dicho tamaño se tuvo en cuenta que la cantidad de variables se duplica porque se utilizaron 2 cláusulas, y que cada variable podía tomar 9 valores diferentes.
El sistema se apoyó en BICIAM y se configuró de la siguiente manera:

- Se ejecutaron 30 veces los algoritmos: Búsqueda Aleatoria, Escalador de Colinas (con Primer Ascenso y aceptando sólo las soluciones mejores) y Algoritmo Genético, haciendo en cada ejecución un número máximo de 500 iteraciones.
- Selección de la mitad de la población (20 individuos) por truncamiento.
- Cruzamiento uniforme con una probabilidad de 0.9
- Mutación en un punto con una probabilidad de 0.5
- Dos cláusulas por predicado
- Operadores de lógica difusa: Min-Max, Media Geométrica y su Dual.

En la tabla 3 se muestran los resultados obtenidos, es decir, los mejores predicados con su nivel de certeza (expresado como valores entre 0 y 1).

Tabla 3. Resultados obtenidos con la propuesta.

Predicado (Min-Max)	**Certeza**
$(I \vee \neg GIP^{0.5} \vee P^{2}) \wedge (I^{0.5} \vee \neg GIP^{0.5})$	0.98
$\neg GIP^{0.5} \vee I$	0.98
$(I^{0.5} \vee \neg GIP^{0.5} \vee P^{0.5}) \wedge (I^{0.5} \vee \neg GIP \vee \neg P^{0.5})$	0.96
$\neg GIP \vee I$	0.96
$\neg GIP^{2} \vee I$	0.93
$(I^{2} \vee \neg GIP \vee P^{2}) \wedge (I \vee \neg GIP^{2} \vee P^{0.5})$	0.92
Predicado (Media Geométrica - Dual)	**Certeza**
$(I^{0.5} \vee \neg GIP^{0.5}) \wedge (I^{0.5} \vee \neg GIP)$	0.98
$I^{0.5} \vee (I^{0.5} \wedge \neg GIP^{0.5})$	0.98
$(I^{0.5} \vee \neg GIP^{0.5}) \wedge (I \vee \neg GIP^{0.5} \vee P^{0.5})$	0.97
$(I^{0.5} \wedge P^{0.5}) \vee I^{0.5}$	0.96
$(I^{0.5} \vee \neg GIP^{0.5}) \wedge (I \vee \neg GIP^{0.5} \vee P^{2})$	0.96
$(I \vee \neg GIP \vee \neg P^{2}) \wedge (I \vee \neg GIP)$	0.95
$I^{0.5} \vee (I \wedge \neg GIP)$	0.94

IV. DISCUSIÓN

Un análisis detallado al interior de los experimentos y predicados mostrados en la tabla 3, aporta los siguientes resultados: el algoritmo obtiene muy buenos resultados con ambos operadores de lógica difusa, teniendo en cuenta que el mayor valor posible a obtener de certeza es 1 y todos están por encima de 0.9. Aunque se considera que se deben incorporar otras métricas que permitan medir la calidad de los resultados obtenidos, el método propuesto obtiene resultados consistentes en todos los casos. Por ejemplo no hay contradicción en ningún predicado de que la inflación es alta, pues aparece sin negar en todos los casos. Por otro lado, la variable GIP aparece negada en todas las cláusulas, asociada fundamentalmente al modificador algo; lo cual indica que el producto interno bruto siempre es bajo, en ambas ejecuciones de los algoritmos con diferentes operadores se logra la presencia de modificadores, característica que marca la diferencia con respecto a otros métodos de descubrimiento de conocimiento. Como debilidad del algoritmo se puede observar una tendencia marcada a la presencia del modificador "algo", ya que su efecto justamente es el aumentar el valor de certeza final. Se podría entonces tratar de reajustar el algoritmo y penalizar su presencia cuando sea excesiva, sara llegar a los resultados anteriormente mostrados no fue necesario disponer de un conjunto de entrenamiento, por lo que se considera que el aprendizaje no supervisado facilita la aplicación del método en diferentes contextos a usuarios no expertos, Los modelos obtenidos muestran diversidad, de un predicado a otro hay cambios en el tipo de conectiva, cantidad de cláusulas y cantidad de variables involucradas (debido al tipo de codificación y operadores utilizados). Los resultados además son fáciles de interpretar pues los predicados no son excesivamente grandes. Aunque se debe velar porque no haya soluciones repetidas y un elemento novedoso incorporado es que no es necesario distinguir cuál algoritmo metaheurístico fue utilizado para obtener cada solución, pues no se busca decidir cuál algoritmo es mejor que otro. Simplemente la propuesta no se limita al uso exclusivo de los tan populares algoritmos evolutivos y lo que realmente se persigue es encontrar buenas soluciones de cara al posterior apoyo a la toma de decisiones por la vía que sean (siendo esto totalmente transparente para el usuario final).

V. CONCLUSIONES

1. El método de extracción de conocimiento propuesto utiliza como medio de codificación predicados en forma normal conjuntiva y disyuntiva, para la evaluación utiliza operadores difusos, y para la exploración del espacio de búsqueda se apoya en un algoritmo metaheurístico; que tiene como finalidad maximizar el valor de verdad de las soluciones obtenidas.

2. Esta propuesta se distingue del resto por varios aspectos: el aprendizaje es no supervisado debido a que no necesita un conjunto de entrenamiento, los predicados pueden cambiar su forma representando distinto conocimiento, se pueden elegir diferentes operadores de lógica difusa para realizar las operaciones clásicas de conjunción y disyunción (tratando de que estas sean lo más apropiado posible para la toma de decisiones) y los algoritmos que se proponen para la búsqueda no se limitan al uso exclusivo de los tan populares algoritmos evolutivos.
3. La propuesta fue aplicada a un problema real de extracción de conocimiento en datos de la economía mexicana. Esta aproximación preliminar al problema permitió obtener conjuntos de predicados con un nivel alto de certeza que fueron fáciles de interpretar por ser un problema pequeño y demostró que aún quedan deficiencias por solucionar. Como trabajo futuro nos planteamos en primer lugar un nuevo estudio experimental sobre una base de ejemplos reconocida internacionalmente y que esté disponible en repositorios como UCI (http://archive.ics.uci.edu/ml/); que permita la comparación del enfoque propuesto con otros que hayan mostrado eficiencia.
4. Además existe la necesidad de integrar la propuesta a una plataforma de minería de datos, como pudiera ser KNIME (Konstanz Information Miner) y así poder hacer uso de las ventajas que esta poderosa herramienta ya brinda. Esto facilitaría sin dudas la implantación de la propuesta y ampliaría su rango de aplicación.

VI. REFERENCIAS

1. HERNÁNDEZ, O.; RAMÍREZ, Q.; FERRI, R., *Introducción a la minería de datos*, Madrid, Pearson Education, 2004, ISBN: 84-205-4091-9.
2. DELGADO, M.; RUÍZ, M.; SÁNCHEZ, D., «Reglas de asociación difusas: Nuevos Retos», en *Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF08)* Cuencas Mineras (Mieres - Langreo), 2008, [consulta: 2011-06-30]. ISBN 978-84-691-5807-4. Disponible en: <http://www.softcomputing.es/estylf08/es/2008-XIV%20Congreso.html>
3. ABRAHAM, A.; GROSAN, C.; RAMOS, V., *Swarm Intelligence in Data Mining. Studies in Computational Intelligence*, vol. 34, Berlin, Springer-Verlag, 2006, ISBN 3-540-33458-0.
4. HONG, T., «Genetic-Fuzzy Data Mining With Divide-and-Conquer Strategy», *IEEE Transactions on Evolutionary Computation* [en línea], 2008, vol. 12, no. 2, pp. 252-265 [consulta: 30-06-2011], ISSN 1089-778X Disponible en: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4358777>
5. HONG, T.; KUO, C.; CHI, S., «Mining association rules from quantitative data. Intelligent Data Analysis», [en línea], 1999, pp. 363-376 [consulta: 14-09-2010], Disponible en: <http://iospress.metapress.com/index/171253281266L437.pdf>
6. MATA, J.; ÁLVAREZ, J.; RIQUELME, J., «Discovering Numeric Association Rules via Evolutionary Algorithm», en *6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining Taipei* Taiwan, 2002, [consulta: ISBN 3-540-43704-5. Disponible en: <http://dblp.uni-trier.de>
7. ZADEH, L., «Fuzzy logic, neural networks, and soft computing», *Communications of the ACM* [en línea], 1994, vol. 37, no. 3, pp. 77-84 [consulta: 15-04-2011], ISSN 0001-0782. Disponible en: <http://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCwQFjAA&url=http%3A%2F%2Fdl.acm.org%2Fcitation.cfm%3Fid%3D175255&ei=8S3dUt2FNKHmsATmg4CADA&usg=AFQjCNGfQ923SasIcKl_UF4gNlrMqcy6ZA&bvm=bv.59568121,d.aWM>
8. ALCALÁ-FERNÁNDEZ, J.; ALCALÁ, R.; GACTO, M. J., «Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms» *Fuzzy Sets and Systems*, 2009, vol. 160, no. 7, pp. 905-921, ISSN 0165-0114.
9. TRILLAS, E., «On a model for the meaning of predicates. A naive approach to the genesis of fuzzy sets. Studies», *Fuzziness and Soft Computing* [en línea], 2009, vol. 243, no. 9, pp. 175-205 [consulta: 20-12-2010], ISSN: 1434-9922. Disponible en: <http://link.springer.com/chapter/10.1007/978-3-540-93802-6_9>
10. JESUS, M. Del, «Extracción de reglas DNF Difusas en un problema de Marketing», en *XII Congreso Español de Tecnologías y lógica Difusa (ESTYLF'04)* Jaen, España, 2004, [consulta: ISBN 84-609-2160-3. Disponible en: <http://estylf04.ujaen.es/>
11. BRUNO, A. D., «Normal forms, Mathematics and Computers in Simulation» 1998, vol. 45, pp. 413-427, ISSN: 0378-4754.

12. MARTÍNEZ, M.; ROSETE, A.; ESPÍN, R., «Experiencias en el descubrimiento de conocimientos a partir de la obtención de predicados en lógica difusa compensatoria», en *Segundo Taller de Descubrimiento de Conocimiento, Gestión del Conocimiento y Toma de Decisiones* Ciudad de Panamá, 2009, [consulta: ISBN 978-959-961-302. Disponible en:
13. RUSSEL, S; NORVIG, P. , *Inteligencia Artificial. Un enfoque moderno*, México, Prentice Hall Hispanoamericana, 1996, ISBN 968-880-682-X.
14. RICH, E.; KNIGHT, K., *Inteligencia artificial*, 2 a. ed., España, McGraw-Hill, 1994, ISBN 84-481-1858-8, p.
15. ESPÍN, R.; LECICH, M., «Decision making and fuzzy inference: a new linked approach», en *Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT)* Barcelona, 2005, [consulta: ISBN 84-7653-872-3. Disponible en: <http://www.informatik.uni-trier.de/~ley/db/conf/eusflat/eusflat2005.html>
16. GAREY, M. R; JOHNSON, D. S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, San Francisco, W.H.Freeman and Company, 1979, ISBN 0716710455.
17. BLUM, C.; ROLI, A., «Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison» *ACM Computing Surveys*, 2003, vol. 35, no. 3, pp. 268-308, ISSN 0360-0300.
18. MELIÁN, B.; MORENO, J.; MORENO, J., «Metaheurísticas: una visión global» *Revista Iberoamericana de Inteligencia Artificial*, 2003, vol. 2, no. 19, pp. 7-28, ISSN 1137-3601.
19. OSMAN, I.; LAPORTE, G., «Metaheuristics: A bibliography», *Annals of Operations Research* [en línea], 1996, vol. 63, no. 5, pp. 511-623 [consulta: 03-11-2010], ISSN 1572-9338. Disponible en: <http://link.springer.com/article/10.1007/BF02125421>
20. TALBI, E., «A Taxonomy of Hybrid Metaheuristics», *Journal of Heuristics* [en línea], 2002, vol. 8, no. , pp. 541-564 [consulta: 16-10-2012], ISSN 1572-9397. Disponible en: <http://dl.acm.org/citation.cfm?id=595102>
21. WOLPERT, D.; MACREADY, W., «No Free Lunch Theorems for Optimization», *IEEE Transactions on Evolutionary Computation* [en línea], 1997, vol. 1, no. 1, pp. 67-82 [consulta: 16-10-2012], ISSN 1089-778X Disponible en: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=585893>
22. FAJARDO, J; SUAREZ A. , , Colombia, 2010, Vol. 1, no. 0, , , «Algoritmo Multigenerador de Soluciones para la competencia y colaboración de generadores metaheurísticos», *Revista Internacional de Investigación de Operaciones* [en línea], 2010, vol. 1, pp. 57-62 [consulta: 16-02-2011], ISSN 2145–9517. Disponible en: <http://www.researchgate.net/publication/232696764_THE_USE_OF_MULTIATTRIBUTE_UTILITY_METHODS_IN_THE_EVALUATION_OF_APPLICANTS_TO_AN_ENGINEERING_SCHOOL_IN_BRAZIL/file/d912f5089cde06ea75.pdf#page=57>