METODOLOGÍA PARA LA GENERACIÓN AUTOMÁTICA DE REGLAS BORROSAS Y AJUSTE ADAPTATIVO DE FUNCIONES DE PERTENENCÍA POR MEDIO DE UNA ARQUITECTURA DE RED NEURONAL NETFUZ 1.0

JUAN CARLOS REYES FIGUEROA
Ingeniero de Sistemas, Candidato a Magíster en Informática
Profesor Auxiliar
Universidad Industrial de Santander
Escuela de Ingeniería de Sistemas, Facultad de Fisicomécanicas
Juankar59@hotmail.com

FERNANDO RUIZ DÍAZ
Ingeniero de Sistemas, Master of Engineering
Profesor Asociado
Universidad Industrial de Santander
Escuela de Ingeniería de Sistemas, Facultad de Fisicomécanicas
fruizd@uis.edu.co

Fecha Recepción: 20 de mayo de 2006
Fecha Aceptación: 21 de noviembre de 2006

RESUMEN

En la generación de los Sistemas de Inferencia Borrosos, la tarea primordial es la extracción y el ajuste de las funciones de pertenencia y las reglas borrosas. Sin embargo, al usar los métodos tradicionales para realizar esta tarea, los resultados obtenidos no son los esperados y en la mayoría de casos se presentan graves inconvenientes. Este artículo presenta una propuesta metodológica basada en Redes Neuronales Artificiales que permite extraer automáticamente las reglas borrosas y los parámetros de las funciones de membresía de un Sistema de Inferencia Borroso tipo Sugeno, partiendo de un conjunto de datos entrada-salida. Se contempla el desarrollo de un software que facilitará la aplicación en el control de procesos, la predicción y la estimación de parámetros.

PALABRAS CLAVE: *Red Neuronal Artificial, Sistema de Inferencia Borroso, Lógica Borrosa, Funciones de Membresía, Sistemas Neuro-Borrosos, COBOR 2.0*

ABSTRACT

In the generation of the Fuzzy Inference Systems, the primordial task is the extraction and the tuning of the memberships functions and the fuzzy rules. However, when using the traditional methods to carry out this task, the obtained results are not the prospective ones and in most of cases serious inconveniences are presented. This article presents a methodological proposal base in Artificial Neural Networks that allows extracting the fuzzy rules and the parameters of the functions of membership of a Fuzzy Inference System type Sugeno automatically, leaving of a group of data input-output. The development of a software is contemplated that will facilitate the application in the control of processes, the prediction and the estimate of parameters.

KEYWORDS: *Artificial Neural Network, Fuzzy Inference Systems, Fuzzy Logic, Membership Functions, Fuzzy-Neural Systems, COBOR2.0*

INTRODUCCIÓN

En la actualidad se revelan nuevas e innovadoras metodologías en la solución de problemas que contienen un alto grado de realismo y para los cuales es imposible establecer un modelo que determine su comportamiento por medio de estrategias convencionales. Es ahí donde La Lógica Borrosa (LB) y, específicamente, Los Sistemas de Inferencia Borrosos, han demostrado ser la alternativa para hallar un resultado que satisfaga las expectativas del experto en procesos.

En suma, este tipo de sistemas soporta el conocimiento y la experiencia de los expertos, describiéndolo, por medio de un conjunto de reglas de situación-acción, las cuales se fundamentan en los conjuntos borrosos; además de facilitar la representación de las entradas y las salidas mediante Funciones de membresía (FM).

Pero ¿Cuántas FM utilizar?, ¿de qué tipo?, ¿qué número de reglas se deben generar?, estos interrogantes se le presentan a menudo a los usuarios de los Sistemas de Inferencia Borrosos, razón por la cual se dificulta obtener una buena descripción de un proceso en particular.

Alrededor de esta situación, se pretende proponer una metodología que genere de manera automática las FM y las reglas borrosas de un Sistema de Inferencia Borroso tipo Sugeno, por medio de una Red Neuronal Artificial Adaptativa, garantizando así una mayor eficacia en los resultados obtenidos.

La presente investigación está soportada en el proyecto de pregrado denominado Cobor 2.0 "Software Para el Control Borroso" que permite trabajar con los Sistemas de Inferencia Borrosos tipo Mandami y tipo Sugeno (ver Figura 1), y se presenta como la base computacional con la cual se pretende llevar a buen término la presente propuesta [1].

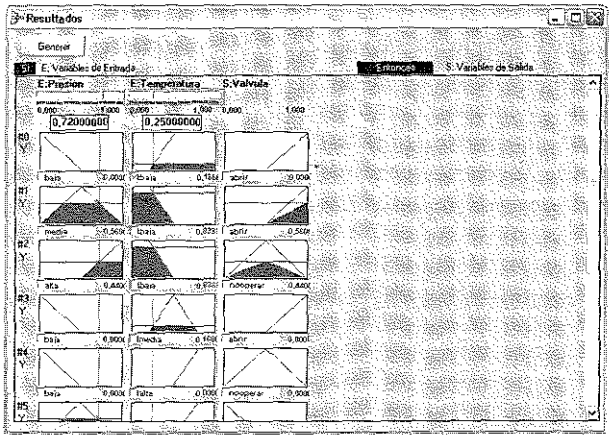

Figura 1. Interfaz de Cobor 2.0

A continuación se exhiben algunos trabajos relacionados con el tema, conjuntamente, el marco teórico y conceptual necesario para el desarrollo e implementación de la metodología propuesta.

TRABAJOS REALIZADOS

En cuanto al ámbito internacional se han desarrollado trabajos de esta naturaleza, se destaca la herramienta computacional ANFIS "Adaptive Networks Based Fuzzy Inference System (1993)" [2], que surgió como el primer intento válido de un Sistema que integra las teorías de La LB y Las Redes Neuronales Artificiales (RNA).

También se ha optimizado la estructura de los Sistemas de Inferencia Borrosos por medio de los algoritmos genéticos y otros métodos de optimización. En lo que tiene que ver con la base de reglas borrosas y el ajuste de las FM, se encuentra un extenso número de trabajos en la literatura [3].

Cabe señalar que la metodología Anfis se implementó con éxito en el Software Matlab, específicamente, en la Fuzzy Logic Toolbox [4] en el año de 1997.

Del mismo modo se ha intentado fusionar La Teoría Borrosa con otras Tecnologías adaptativas como: Los Agentes Inteligentes y el recocido simulado, en procura de ampliar el campo de aplicación [5].

A pesar de los avances significativos mostrados a nivel internacional, en Colombia es limitado el desarrollo de herramientas de este tipo, pero se abona el interés despertado en los últimos años, reflejado en el número de aplicaciones desarrolladas por los diferentes entes educativos [6].

SISTEMAS DE INFERENCIA BORROSOS

Se denomina inferencia al proceso de extraer una conclusión a partir de ciertas premisas y un conjunto de reglas, Los Sistemas de Inferencia Borrosos (SIB), son una plataforma popular de cómputo soportada en los conceptos de La LB, las reglas borrosas If – Then, y el razonamiento borroso.

La LB se basa en la Teoría de Conjuntos que posibilita imitar el comportamiento de la Lógica Humana. El término «borroso» procede de la palabra inglesa «Fuzzy» que significa «confuso, difuso, indefinido o desenfocado». Hace parte de La Inteligencia Artificial, y se funda en el concepto «Todo es cuestión de cuanto se cumple", lo cual permite manejar información vaga o de difícil especificación [7].

Es así como es posible con la LB gobernar un sistema por medio de reglas de sentido común, las cuales se refieren a

cantidades indefinidas. Este tipo de reglas pueden ser desarrolladas con sistemas adaptativos, que aprenden al 'observar' como operan las personas los dispositivos reales o, simplemente, ser formuladas por un experto humano.

En general, la LB se aplica tanto a sistemas de control como para modelar cualquier sistema continuo de Ingeniería, Física, Biología o Economía, definiendo así, un sistema matemático que modela funciones no lineales, convirtiendo las entradas en salidas acordes con los planteamientos lógicos que usan el razonamiento aproximado.

Cabe señalar que la LB se fundamenta en la Teoría de los Conjuntos Borrosos, (ver Figura 2), y las reglas borrosas de la forma «If [Condición] Then [Acción]", donde los valores lingüísticos de la premisa y el consecuente están definidos por FM [8].

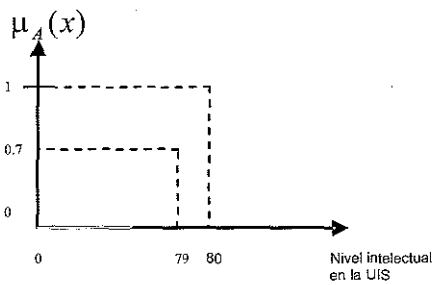

Figura 2. Conjunto Borroso

En resumidas cuentas la LB es un área fascinante de la investigación, debido a que proporciona un soporte formal al razonamiento basado en el lenguaje natural, además de compaginar significancía y precisión. En la Figura 3 se observa un claro ejemplo de lo que significan estas dos palabras.

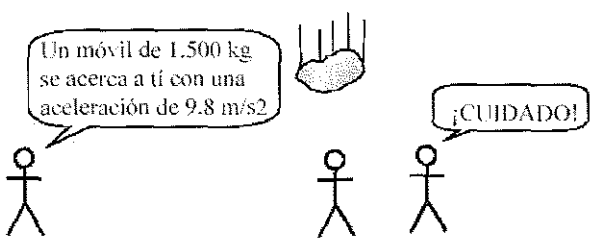

Figura 3. Significancia y Precisión

La estructura básica de un SIB contiene cinco componentes conceptuales (ver Figura 4):

* **La Fuzzificación:** Convierte la entrada lógica o crisp en un valor borroso determinado por una función de pertenencia.

* **La Base de Datos:** Define las funciones de pertenencia que se utilizan en las reglas borrosas.

* **La Base de Reglas:** Contiene el conjunto de acciones a realizar en función del estado.

* **El Mecanismo de Inferencia:** Realiza el procedimiento de inferencia sobre las reglas borrosas y los hechos o datos suministrados para producir una respuesta o conclusión.

* **La Defuzzificación:** Proceso que consiste en extraer u obtener un valor binario de un conjunto borroso de salida como valor representativo, utilizando los métodos existentes para ello.

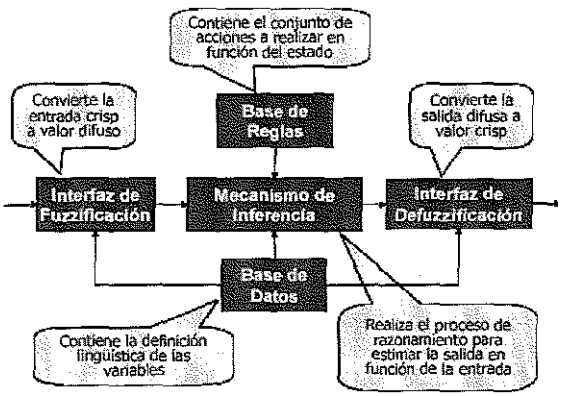

Figura 4. Sistema de Inferencia Borroso [9]

Tipos de Sib

Tipo Mandami

El SIB tipo Mandami fue propuesto en el año de 1975 por Ebrahim Mandami y un grupo de estudiantes del Colegio Queen Mary de Londres, como el primer intento para controlar un motor de vapor en combinación con una caldera mediante un conjunto de reglas lingüísticas adquiridas de operadores humanos con experiencia, donde se obtenía un conjunto borroso de salida al cual se le aplicaba un método de defuzzificación [10].

Las reglas de un SIB tipo Mandami poseen la siguiente estructura:

if x is small then z is small (1)

Tipo Sugeno
También conocido como el modelo TSK, fue propuesto por Takagi, Sugeno y Kang en el año de 1985 como un esfuerzo para desarrollar un enfoque sistemático que genera las reglas borrosas a partir de un conjunto de datos entrada – salida [11].

Una regla borrosa típica en un modelo de Sugeno tiene la forma:

$$if\ x\ is\ small\ then\ z = -x + 1 \qquad (2)$$

Donde $z = -x + 1$ es un conjunto borroso y una función crisp; si la función z es polinomial se tiene un sistema de primer orden, en el caso de ser z una constante se tiene un sistema de orden cero.

Tipo Tsukamoto
En este modelo el consecuente de cada regla borrosa If – Then se representa por un conjunto borroso con una función de membresía monótona. Como consecuencia la salida de cada regla se define como un valor binario inducido por la fuerza del disparo de la regla. La salida total se toma como el peso promedio de salida de cada una de las reglas.

Una regla borrosa típica en un modelo de Tsukamoto tiene la forma:

$$if\ x\ is\ small\ then\ z = C_1 \qquad (3)$$

Redes Adaptativas
Las arquitecturas y los tipos de aprendizaje de las redes adaptativas, son de hecho un superconjunto de todas las clases de Redes Neuronales Artificiales tipo Feedforward, con capacidad de aprendizaje supervisado. Este tipo de redes contiene una estructura conformada por nodos los cuales están interconectados. Más aún todos o parte de los nodos son adaptativos y dependen de parámetros iniciales . La regla de aprendizaje que aplica sobre estos nodos debe estar cambiando en pos de minimizar el error global.

La regla básica de aprendizaje de las redes adaptativas se basa en el gradiente descendente y la regla de la cadena, que fue propuesta por Werbos en el año de 1970 [12].

Arquitectura y Regla de Aprendizaje Básica de Una Red Adaptativa

Una red adaptativa (ver figura 5), es una red multicapa con conexiones hacia delante, en donde cada nodo lleva a cabo una función particular sobre las señales que llegan a él. Las funciones dependen de un conjunto de parámetros propios de cada nodo.

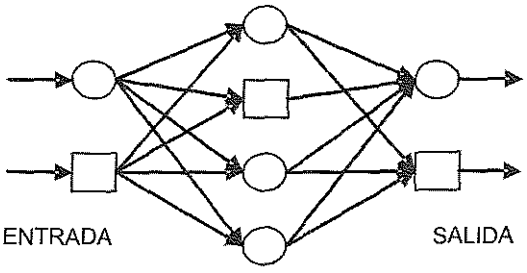

Figura 5. Arquitectura de una Red Adaptativa

Para reflejar las capacidades adaptativas de los nodos, se usaron círculos y cuadrados para diferenciarlos, dentro de las redes adaptativas. El cuadrado es un nodo adaptativo que posee parámetros, mientras que el círculo es un nodo fijo y por ende no posee parámetros. El conjunto de parámetros de una red adaptativa es la unión de los parámetros de todos los nodos adaptativos.

Los parámetros de los nodos adaptativos se derivan del conjunto de datos de entrenamiento y son actualizados por medio de un aprendizaje basado en el gradiente, este proceso es descrito a continuación:

Suponga que se tiene una red adaptativa con L capas y la k - *th* capa tiene *#(k) nodos*. Es posible denotar el nodo de la i - *th* posición en la k - *th* capa por *(k,i)*, y la función de nodo (o salida del nodo) por O^k_i. Donde la salida del nodo depende de las señales de entrada y el conjunto de parámetros propios del nodo:

$$O_i^k = O_i^k(O_i^{k-1},...,O_{\#(k-1)}^{k-1}, a, b, c, ...) \qquad (4)$$

Donde $a,b,c...$ son los parámetros pertenecientes a este nodo. Asumiendo que el conjunto de datos de entrenamiento dado tiene P entradas, es posible definir el error medio (o función energía) para la p - *th* $(1 \le p \le P)$ entrada de los datos de entrenamiento, como la suma de los errores cuadrados:

$$E_p = \sum_{m=1}^{\#L} (T_{m.p} - O_{m.p}^L)^2 \qquad (5)$$

Donde $T_{m.p}$ es el m - *th* componente de p - *th* vector de salida deseada, y $O^L_{m.p}$ es el m - *th* del actual vector de

salida producido por la presentación del p - th vector de entrada. El error medio total es dado por

$$E = \sum_{p=1}^{P} E_p$$

Siguiendo el orden de desarrollo del procedimiento de aprendizaje, se debe implementar el método del gradiente descendente en E sobre el espacio de parámetros; primero se calcular la rata de error $\frac{\partial E_p}{\partial O}$ para el p - th dato de entrenamiento y para cada nodo de salida O. La rata de error para un nodo de salida como (L,i) puede ser calculada directamente con la ecuación (5):

$$\frac{\partial E_p}{\partial O_{i,p}^L} = -2(T_{i,p} - O_{i,p}^L) \qquad (6)$$

Para los nodos internos como (k,i), la rata de error es derivada por la regla de la cadena:

$$\frac{\partial E_p}{\partial O_{i,p}^k} = \sum_{m=1}^{\#(k+1)} \frac{\partial E_p}{\partial O_{m.p}^{k+1}} \frac{\partial O_{m.p}^{k+1}}{\partial O_{i,p}^k} \qquad (7)$$

Donde $1 \le k \le L-1$, esto asegura que la rata de error de un nodo interno puede ser expresada como una combinación lineal de las ratas de error de los nodos de la siguiente capa. Por lo tanto para todo $1 \le k \le L$ y $1 \le i \le \#(k)$ es posible encontrar $\frac{\partial E_p}{\partial O_{i,p}^k}$, con las ecuaciones (6) y (7). Ahora si α es un parámetro de la red adaptativa dada, entonces:

$$\frac{\partial E_p}{\partial \alpha} = \sum_{O^* \in S} \frac{\partial E_p}{\partial O^*} \frac{\partial O}{\partial \alpha} \qquad (8)$$

Donde S es el conjunto de nodos cuya salida depende de α. Entonces la derivada del error medio total con respecto a α es:

$$\frac{\partial E}{\partial \alpha} = \sum_{p=1}^{P} \frac{\partial E_p}{\partial \alpha} \qquad (9)$$

Por consiguiente, la formula de actualización para el parámetro genérico α es:

$$\Delta\alpha = -\eta \frac{\partial E}{\partial \alpha} \qquad (10)$$

En la cual η es la rata de aprendizaje que puede ser expresada como:

$$\eta = \frac{k}{\sqrt{\sum_{\alpha} \left(\frac{\partial E}{\partial \alpha}\right)^2}} \qquad (11)$$

Donde k es la medida del paso, para variar la velocidad de convergencia del gradiente es necesario aumentar o disminuir el paso.

Actualmente, existen dos paradigmas de aprendizaje para redes adaptativas, en el aprendizaje por lotes (o aprendizaje Off-Line), la formula de actualización para los parámetros α se basan en la ecuación (9) y la acción de actualización toma lugar sólo después de que el conjunto de datos de entrenamiento completo ha sido presentado. Por otro lado si se quieren actualizar los parámetros de manera inmediata después de cada par entrada-salida presentado, la formula de actualización es basada en la ecuación (8) y esta se refiere a un aprendizaje por patrones (o aprendizaje On-Line).

Regla de Aprendizaje Hibrido: Aprendizaje por Lotes (off-line)

Lo común es aplicar el método del gradiente para identificar los parámetros en una red adaptativa, pero este método, generalmente, es lento y queda atrapado con facilidad en un mínimo local. Lo que se propone es una regla de aprendizaje hibrido [13], que combina al método del gradiente con el estimador de mínimos cuadrados (LSE) para identificar los parámetros de la red adaptativa.

Por simplicidad se asume que la red adaptativa bajo consideración tiene una salida:

$$output = F(\vec{I}, S) \qquad (12)$$

Donde $\vec{I}$ es el conjunto de variables de entrada y S es el conjunto de parámetros. Si existe una función H tal que la composición de funciones $H \circ F$ sea lineal en alguno de los elementos de S, entonces estos elementos pueden ser identificados por el método de mínimos cuadrados, formalmente, si el conjunto de parámetros puede ser descompuesto en dos conjuntos:

$$S = S_1 \oplus S_2 \qquad (13)$$

(Donde $\oplus$ representa la suma directa) tal que $H \circ F$ es lineal en los elementos de S_2, entonces sobre H se aplica la ecuación (12), que define:

$$H(output) = H \circ F(\vec{I}, S) \qquad (14)$$

Que es lineal en los elementos de S_2. Luego se proporcionan valores de elementos de S_1, ahora es posible presentar los datos de entrenamiento P en la ecuación (14) y obtener una formula matricial:

$$AX = B \qquad (15)$$

Donde X es un vector desconocido cuyos elementos son parámetros de S_2. Al permitir que $|S_2| = M$, entonces las dimensiones de A,X y Bson: $P \times M, M \times 1$ y $P \times 1$

respectivamente. Desde que P (número de pares de datos de entrenamiento) sea mayor que M (número de parámetros lineales), este problema se convierte en sobredeterminado y no es posible encontrar una solución exacta a la ecuación (15). Es preferible utilizar el Estimador de Mínimos Cuadrados (LSE) de X,X^*, si se busca minimizar el error cuadrado $\|AX - B\|^2$, al final queda convertido en un problema estándar que es formulado en regresión lineal, filtros adaptativos y procesamiento de señales. La formula más conocida para hallar X^*es usar la seudo inversa de X:

$$X^* = (A^T A)^{-1} A^T B \qquad (16)$$

Donde A^T es la transpuesta de A, y $(A^T A)^{-1} A^T$ es la seudo inversa de A, si $(A^T A)$ es no singular. Mientras la ecuación es concisa en la notación, esta es muy costosa en cálculos computacionales, más si se trata de encontrar la inversa de la matriz, y sobre todo si $(A^T A)$ es singular. Para evitar este proceso se propone emplear formulas secuenciales para computar los LSE de X.

El método secuencial de LSE es más eficiente (especialmente cuando M es pequeño) y puede ser modificado con facilidad en una versión On-Line para sistemas con características cambiantes. Específicamente, la *i-th* fila de la matriz A definida en la ecuación (15) es a^T_i y el *i-th* elemento de B es b^T_i, entonces X puede ser calculada de forma iterativa usando formulas secuenciales adoptadas ampliamente en la literatura [14], [15], [16]:

$$X_{i+1} = X_i + S_{i+1} a_{i+1} (b^T_{i+1} - a^T_{i+1} X_i)$$
$$S_{i+1} = S_i - \frac{S_i a_{i+1} a^T_{i+1} S_i}{1 + a^T_{i+1} S_i a_{i+1}} \qquad (17)$$

Donde S_i es a menudo llamada matriz de covarianza y el estimador de mínimos cuadrados X^* es igual a X_p. Las condiciones iniciales de la ecuación (17) son: X_o y $S_o = \gamma I$, donde γ es un número positivo grande y I es la matriz identidad de dimensión MxM. Cuando se trabaja con redes adaptativas multi – salida, (el en la ecuación (12) se convierte en un vector columna), es posible aplicar la ecuación (17), excepto cuando b^T_i sea el *i-th* fila de la matriz B.

De lo anterior se concluye que es probable combinar el método del gradiente con el estimador de mínimos cuadrados en una red adaptativa. Cada época de este procedimiento de aprendizaje hibrido es compuesta por un paso hacia atrás y un paso hacia delante. En el paso hacia delante, son suministrados los datos de entrada y las señales funcionales, para calcular la salida de cada nodo hasta que las matrices A y B son obtenidas, y los parámetros en S_2 son identificados por la formula de mínimos cuadrados secuenciales (17). Después de identificar los parámetros en S_2, las señales funcionales se propagan hacia delante hasta que el error medio es calculado. En el paso hacia atrás, la rata de error es propagada desde la salida hasta la entrada, y los parámetros de son actualizados por el método del gradiente según la ecuación (10).

Regla De Aprendizaje Hibrido: Aprendizaje Por Patrones (on-line)

Si los parámetros son actualizados después de cada presentación de datos, se tiene un aprendizaje por patrones o aprendizaje en línea. Este paradigma de aprendizaje es primordial para la identificación de parámetros en línea en sistemas con características cambiantes. Para la formula de mínimos cuadrados secuenciales se debe tener en cuenta las características variables de los datos de entrada. Es necesario que decaiga el efecto de los viejos pares de datos cuándo los nuevos pares de datos se presenten. Un método simple para solucionar el problema es adicionar un factor λ a la formula secuencial original, que le de prioridad a los nuevos datos presentados, según su relevancia dentro del sistema:

$$X_{i+1} = X_i + S_{i+1} a_{i+1} (b^T_{i+1} - a^T_{i+1} X_i)$$
$$S_{i+1} = \frac{1}{\lambda} [S_i - \frac{S_i a_{i+1} a^T_{i+1} S_i}{\lambda + a^T_{i+1} S_i a_{i+1}}] \qquad (18)$$

Donde el valor de λ está entre 0 y 1.

Netfuz 1.0

Dado el alcance obtenido por las Tecnologías Adaptativas en el ámbito nacional e internacional, surgió la necesidad de su estudio e implementación, para incrementar el interés por generar aplicaciones y herramientas que permitieran el desarrollo de estás teorías y su aplicabilidad.

Netfuz 1.0 surge como una herramienta basada en redes adaptativas, descritas con antelación, que aprovechan su carácter flexible para implementar una equivalencia con los SIB tipo Sugeno, y cuyo propósito es la generación automática de reglas borrosas y el ajuste de los parámetros en las FM. Apoyando su fundamento en la regla de aprendizaje hibrido, y tomando de referencia autores como: Kosko, Zadeh y Jang, que determinaron los principios básicos de los sistemas Neuro-Borrosos, los cuales combinan las bondades propias de las RNA y la LB [17].

- Tener una salida única, cuyo fuzzificación será obtenida

PROPUESTA METODOLÓGICA ARQUITECTURA Y APRENDIZAJE DE NETFUZ 1.0

Arquitectura Netfuz 1.0

Por simplicidad, se asume el SIB bajo consideración con dos entradas x y y, y una salida z. Se supone que la base de reglas contiene dos reglas borrosas If-Then, del modelo Sugeno:

$$si\ x\ es\ A_1\ Y\ y\ es\ B_1\ entonces\ f_1 = p_1x + q_1y + r_1$$
$$si\ x\ es\ A_2\ Y\ y\ es\ B_2\ entonces\ f_2 = p_2x + q_2y + r_2$$

Se asumen los consecuentes de las reglas con tres parámetros lineales. El razonamiento borroso tipo Sugeno es ilustrado en la figura 6(a), y su equivalencia en la arquitectura **Netfuz 1.0** es exhibido en la figura 6(b).

A continuación se definen las funciones y procedimientos a realiza según cada una de las capas que componen la arquitectura.

Capa 1: cada nodo i de esta capa es un nodo cuadrado con una salida de la forma:

$$O_i^1 = \mu_{A_i}(x) \tag{19}$$

Donde x es la entrada al nodo i, y A_i es la etiqueta lingüística (baja, media, etc.) asociada con la salida del nodo. En otras palabras O_i^1 es la FM de A_i. Usualmente, se escoge para $\mu_{A_i}(x)$ una función Bell-shaped, con máximo igual a 1 y mínimo igual a 0, tal que la función bell quede generalizada de la siguiente forma:

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_1}} \tag{20}$$

Donde $\{a_i, b_i, c_i\}$ es el conjunto de parámetros. Como los valores de estos parámetros cambian, las funciones Bell-

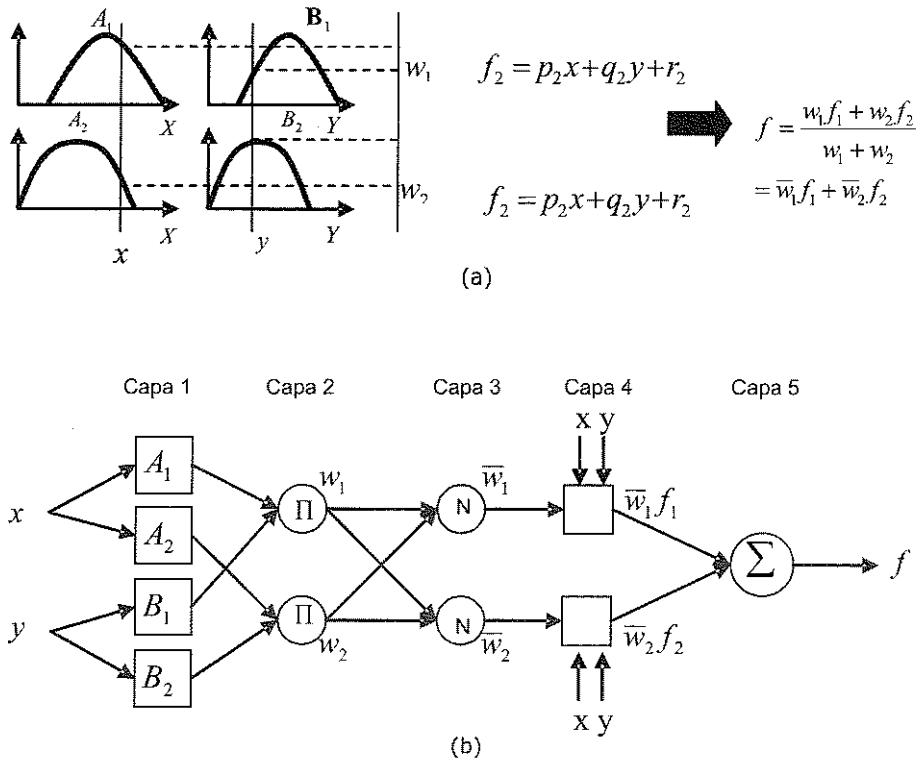

Figura 6. (a) Razonamiento Borroso Tipo Sugeno; (B) equivalente con Netfuz 1.0

shaped por consiguiente varían, de esta manera se exhiben varias formas de funciones de membresía para las etiquetas lingüísticas A_i. Los parámetros de esta capa se denominan parámetros de la premisa y son no-lineales.

Capa 2: Los nodos de esta capa son nodos circulares y se denotan por el símbolo $\prod$, en ellos se multiplican las señales de entrada y la salida del nodo es un producto, como el mostrado en la siguiente ecuación:

$$w_i = \mu_{A_i}(c) \times \mu_{B_i}(y), i = 1,2 \qquad (21)$$

La salida de los nodos de la capa 2 representa la fuerza de disparo de la regla o la implicación.

Capa 3: Los nodos de esta capa son nodos circulares y se denotan por el símbolo N, en el se calcula la suma de todas las fuerzas de disparo de las reglas de la capa anterior:

$$\overline{w}_i = \frac{w_i}{w_1 + w_2} \qquad (22)$$

Por conveniencia, en las salidas de esta capa son normalizadas las fuerzas de disparo de las reglas.

Capa 4: Cada nodo i de esta capa es un nodo cuadrado con una salida de nodo descrita como:

$$O_i^4 = \overline{w}_i f_i = \overline{w}_i (q_i x + p_i y + r_i) \qquad (23)$$

Donde $\overline{w}_i$ es la salida de la capa 3, y $\{p_i, q_i, r_i\}$ es el conjunto de parámetros. Los parámetros de esta capa se denominan parámetros del consecuente y son lineales.

Capa 5: El único nodo de esta capa es un nodo circular denotado por el símbolo $\sum$ aquí se computa la salida total, como la suma de todas las señales de entrada:

$$O_i^5 = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \qquad (24)$$

De esta manera se puede construir una red adaptativa que equivale en funcionamiento a un SIB tipo Sugeno, las extensiones para los SIB tipo mandami y Tsukamoto son viables, pero su complejidad es mayor.

Algoritmo de Aprendizaje Híbrido Netfuz 1.0

De la arquitectura propuesta en la figura 4(a), se observa que se presentan los valores de los parámetros de la premisa como los parámetros de las FM, la salida total puede ser expresada como una combinación lineal de los parámetros del consecuente. Para ser más precisos, la salida f de la Figura 4 puede ser reescrita como:

$$\begin{aligned} f &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 \\ &= \overline{w}_1 f_1 + \overline{w}_2 f_2 \\ &= (\overline{w}_1 x) p_1 + (\overline{w}_1 y) q_1 + (\overline{w}_1) r_1 + (\overline{w}_2 x) p_2 + (\overline{w}_2 y) q_2 + (\overline{w}_2) r_2 \end{aligned} \qquad (25)$$

Que es lineal para los parámetros del consecuente $(p_1, q_1, r_1, p_2, q_2, r_2)$, como resultado se tiene que:

S: Conjunto total de parámetros
S_1: Conjunto de parámetros de la premisa
S_2Conjunto de parámetros del consecuente

De la ecuación (13); $H(.)$ es una función identidad y $F(.,.)$es una función propia de los SIB. Por lo tanto en algoritmo de aprendizaje hibrido desarrollado con antelación puede ser aplicado directamente. En el paso hacia delante del algoritmo de aprendizaje hibrido, las señales funcionales se propagan hasta la capa 4 y los parámetros del consecuente son identificados por medio del estimador de mínimos cuadrados. En el paso hacia atrás, la rata de error se propaga hacia atrás y los parámetros de la premisa son actualizados por el gradiente descendente. La tabla 1 resume las actividades por cada paso:

Tabla 1. Pasos del Aprendizaje Hibrido

	Paso adelante	Paso atrás
Parámetros de la premisa	Fijos	Gradiente descendente
Parámetros consecuente	LSE	Fijos
Señales	Salida de los nodos	Rata de error

Los sistemas con Múltiples entradas y salidas son un caso especial para tratar, aunque es posible implementar la metodología propuesta en ellos, los resultados no son muy halagadores, esto se hace más evidente al aumentar la complejidad de problema planteado.

A continuación se muestran algunas limitaciones de **Netfuz 1.0**:

- Apto sólo para sistemas Sugeno de orden 0 (el consecuente es un número real) ó 1 (el consecuente es un polinomio de primer grado).

por el método de la media de pesos, todos los consecuentes de las reglas deben ser del mismo tipo ya sea de tipo lineal o de tipo constante.

- El número de reglas del Sistema de Inferencia Borroso es igual al producto de las FM de las entradas del sistema.

- Todas las reglas deben tener un peso igual a 1.

Ejemplo de Aplicación

Para probar la herramienta se tomaron 25 datos experimentales de entrada/ salida que validan la operación $z = \sin(x) * \cos(x)$ definida para el intervalo $[-\pi/2, \pi/2]$, y se muestran en la tabla 2.

Tabla 2. Datos Experimentales

iteración	X	Y	Z
1	-1.57079	-1.57079	0
2	-1.57079	-0.94247	-0.587785
3	-1.57079	-0.314159	-0.951056
4	-1.57079	0.314159	-0.951056
5	-1.57079	0.94247	-0.587785
6	-0.94247	-1.57079	0
7	-0.94247	-0.94247	-0.475528
8	-0.94247	-0.314159	-0.769420
9	-0.94247	0.314159	-0.769420
10	-0.94247	0.94247	-0.475528
11	-0.314159	-1.57079	0
12	-0.314159	-0.94247	-0.181635
13	-0.31415 9	-0.314159	-0.293892
14	-0.314159	0.314159	-0.293892
15	-0.314159	0.94247	-0.181635
16	0.314159	-1.57079	0
17	0.314159	-0.94247	0.181633
18	0.314159	-0.314159	0.293892
19	0.314159	0.314159	0.293892
20	0.314159	0.94247	0.181635
21	0.94247	-1.57079	0
22	0.94247	-0.94247	0.475528
23	0.94247	-0.314159	0.769420
24	0.94247	0.314159	0.769420
25	0.94247	0.94247	0.475528

El ejercicio de aplicación contiene dos entradas y una salida, las entradas poseen tres particiones borrosas: {baja, media, alta}. Al utilizar la metodología propuesta se obtienen los parámetros de las FM para el universo de discurso (o rango), además de las reglas borrosas con sus respectivos consecuentes.

Funciones de Membresía

La función de membresía para el intervalo {bajo} de la variable X es:

$$e^{-\frac{(x+1.561)^2}{0.5422}}$$

La función de pertenencia para el intervalo {medio} de la variable X es:

$$e^{-\frac{(x+0.312)^2}{0.521}}$$

La función de pertenencia para el intervalo {alto} de la variable X es:

$$e^{-\frac{(x-0.932)^2}{0.5475}}$$

La función de pertenencia para el intervalo {bajo} de la variable Y es:

$$e^{-\frac{(x+1.575)^2}{0..5355}}$$

La función de pertenencia para el intervalo {medio} de la variable Y es:

$$e^{-\frac{(x+0.316)^2}{0.547}}$$

La función de pertenencia para el intervalo {alto} de la variable Y es:

$$e^{-\frac{(x-0.949)^2}{0.53}}$$

Reglas Borrosas

Las reglas borrosas obtenidas para el ejercicio surgen como la combinatoria de las FM de las entradas, y se presentan a continuación:

1 *si X es bajo y Y es bajo entonces* $Z = 0.06321$
2 *si X es medio y Y es bajo entonces* $Z = -1.201$
3 *si X es alto y Y es bajo entonces* $Z = -0.6405$
4 *si X es bajo y Y es medio entonces* $Z = 0.02196$
5 *si X es medio y Y es medio entonces* $Z = -0.4172$
6 *si X es alto y Y es medio entonces* $Z = -0.2225$
7 *si X es bajo y Y es alto entonces* $Z = -0.0524$
8 *si X es medio y Y es alto entonces* $Z = 0.9952$
9 *si X es alto y Y es alto entonces* $Z = 0.5309$

Para el ejemplo planteado, los resultados obtenidos por **Netfuz 1.0** al utilizar la regla de aprendizaje hibrido, generaron un error mínimo en un número de iteraciones bajo, superando los resultados obtenidos con el aprendizaje por el método del gradiente descendente, los resultados se muestran en la tabla 3.

Tabla 3. resultados obtenidos

MÉTODO	ERROR	ÉPOCAS
Hibrid o	0.052838	3
Gradiente Des.	0.45035	10
Gradiente Des.	0.004056	100

A partir de los resultados se evidencia que al usar la regla de aprendizaje hibrido, se reduce ostensiblemente el error final en un número de épocas muy pequeño, resultados que contrastan con el método del gradiente que alcanzo un error aceptable pero, en un número de iteraciones muy alto.

La metodología de **Netfuz 1.0** presenta complicaciones cuando la complejidad del problema aumenta, en el caso de cuatro entradas o más, luego se recomienda para ejercicios con dos y tres entradas.

Los campos de aplicación de la herramienta **Netfuz 1.0**, van desde la identificación de sistemas no lineales, la predicción de parámetros, el pronóstico y los sistemas de control, ya sean en línea o no.

Ambiente Computacional

La herramienta desarrollada está estructurada en dos partes: la primera, atañe a la implementación de los SIB tipo Mandami y Sugeno, que corresponde al proyecto Cobor 2.0, desarrollado en el año 2005, bajo un ambiente de programación Delphi 7.0, el cual se implementó bajo la Metodología de Lenguaje Unificado (UML), con el fin de facilitar posibles cambios o evoluciones.

La segunda parte, utiliza el proyecto Cobor 2.0 como base computacional para implementar la metodología propuesta, integrando los SIB convencionales con las redes adaptativas, para generar de manera automática las reglas borrosas y las FM, la finalidad de esta integración es obtener una herramienta computacional integral, que permita trabajar los dos enfoques, a la cual se le denominó Netfuz 1.0 (ver Figura 7).

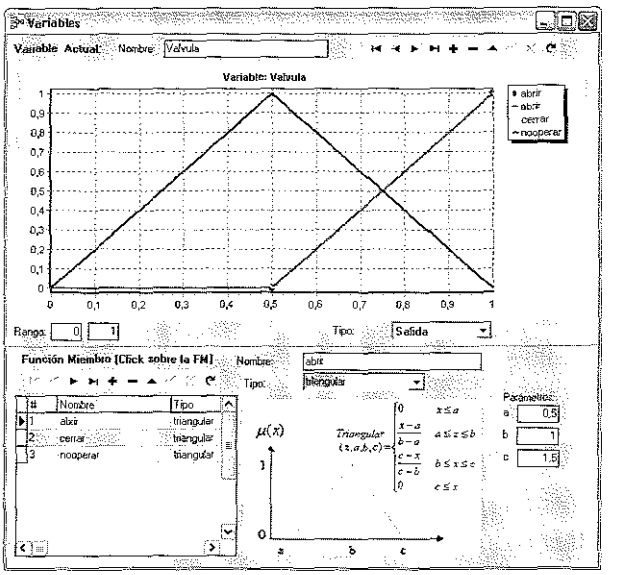

Figura 7. interfaz de Netfuz 1.0

Requerimientos de Hardware y Software

Para el desarrollo de la herramienta Netfuz 1.0 se utilizó un equipo con las siguientes características:

- Procesador Pentium IV de 2.8 Mhz.
- 128 MB de memoria en Ram

La ejecución del sistema exige mínimo tener 128 MB de memoria Ram y un procesador Pentium III o superior. Esta herramienta funciona en plataformas de Windows series 9X, NT, 2000, XP.

Conclusiones y Trabajos Futuros

La simbiosis entre la LB y las RNA, deja al descubierto un gran potencial que se debe explotar en todos los campos de la Ciencia y la Ingeniería.

Además es posible integrar la LB con otras teorías como: los Algoritmos Genéticos y los Agentes Inteligentes, que aportan un ingrediente importante en pos de mejorar las soluciones formuladas en la actualidad.

La metodología propuesta facilita la implementación de los SIB tipo Sugeno, y acrecienta el interés de parte de los investigadores para generar soluciones efectivas que conlleven el uso de Tecnologías Adaptativas.

La RNA adaptativa utilizada presentó buenos resultados en cuanto al algoritmo de aprendizaje hibrido. Se pretende probar otro tipo de aprendizaje, pero este trabajo se deja como investigación a posteriori.

Bibliografía

[1] COBOR 2.0, "Herramienta Software para el Control Borroso", Universidad Industrial de Santander, 2005, Bucaramanga.

[2] JANG J.S.R., "ANFIS: Adaptive Network Based Fuzzy Inference System," IEEE Trans on SMC, 23(3), 665-685, 1993.

[3] NOZAKI K., ISHIBUCHI H., and TANAKA H., "A simple but powerful heuristic method for generation fuzzy rules from numerical data," Fuzzy Sets Syst., vol. 86, pp. 251-270, 1997.

[4] MathWorks, MATLAB Fuzzy Logic Toolbox, the MathWorks Inc., 1997.

[5] LINKENS D.A. and NYONGESA H.O., "Learning systems in intelligent control: On appraisal of fuzzy, neural and genetic algorithm control applications," in Proc. Inst. Elect. Eng. Control Theory Applications, vol. 143, 1996, pp. 367-386.

[6] PEÑA C., Ponencia Coevolutionary Fuzzy Modeling, III Congreso Internacional de Inteligencia Computacional, Universidad del Sinú Colombia, Agosto 10 al 12 de 2005.

[7] ZADEH L., "Fuzzy and sets "Inf, control, vol. 8, pp 338-353, 1965.

[8] HABER R., Introducción al control borroso, 1er seminario internacional sobre control inteligente de procesos, Medellín Colombia, Marzo 6 al 10 de 1995.

[9] PEÑA C., Coevolutionary Fuzzy Modeling, PhD thesis, École Polytechnique Fédérale De Lausanne, Switzerland,2002.

[10] MANDAMI E. H., "Applications of fuzzy logic to approximate reasoning using linguistic Systems". IEEE Trans. On Systems, Man, and Cybernetics, 26(12), pp. 1182-1191, 1977.

[11] TAKAGI T., and SUGENO M., "Fuzzy Identification of System and its Application to Modelling and Control", IEEE Trans on SMC, 15(1), 116-132, 1985.

[12] WERBOS P., beyond regression: New tools for prediction and analysis in the behavioral sciences. PhD thesis, Harvard University, 1974.

[13] JANG J.S.R., Fuzzy modeling using generalized neural networks and Kalman filter algorithm. In Proc. bof the Ninth National Conference on Artificial Intelligence (AAAI-91), pp. 762–767, July 1991.

[14] GOODWIN G. C. and K. SIN S., Adaptive filtering prediction and control. Prentice-Hall, Englewood Cliffs, N.J., 1984.

[15] LJUNG L., System identification: theory for the user. Prentice-Hall, Englewood Cliffs, N.J., 1987.

[16] STROBACH P., Linear prediction theory: a mathematical basis for adaptive systems. Springer-Verlag, 1990.

[17] KOSKO B., Neural Nets y Fuzzy System, prentice Hall, 1992.