Sistemas de lógica difusa. Fundamentos

Oscar G. Duarte V.*

RESUMEN

El propósito de este artículo es presentar los fundamentos de los sistemas de lógica difusa, partiendo de la teoría de conjuntos difusos, hasta la exposición preliminar de las estrategias de entrenamiento. Se presenta también una herramienta de software (UNFUZZY) desarrollada en el Departamento de Ingeniería Eléctrica de la Universidad Nacional de Colombia. El artículo está dirigido a ingenieros no conocedores del tema, y es de caracter divulgativo general.

INTRODUCCIÓN

Desde su aparición en la decada del 60 hasta nuestros días, las aplicaciones de la lógica difusa han ido consolidándose, paulatinamente al comienzo, y con un desbordado crecimiento en los últimos cinco años. Se encuentran en soluciones a problemas de control industrial, en predicción de series de tiempo, como metodologías de archivo y búsqueda de bases de datos, en investigación operacional, en estrategias de mantenimiento predictivo y en otros campos más.

Las principales razones para tal proliferación de aplicaciones quizá sean la sencillez conceptual de los sistemas basados en lógica difusa, su facilidad para adaptarse a casos particulares con pocas variaciones de parámetros, su habilidad para combinar en forma unificada expresiones lingüísticas con datos numéricos, y el no requerir algoritmos muy sofisticados para su implementación.

El propósito fundamental de este artículo es presentar los fundamentos de los sistemas basados en lógica difusa, con la certeza de que algunos de los lectores encontrarán en ellos una alternativa más para abordar sus problemas específicos.

I. TEORÍA DE CONJUNTOS DIFUSOS

Una buena estrategia para presentar la teoría de conjuntos difusos consiste en recordar algunos aspectos de la teoría de conjuntos convencionales (que llamaremos *conjuntos concretos*), y a partir de allí hacer una extensión a los conjuntos difusos:

Un conjunto concreto se define como una colección de elementos que existen dentro de un universo. Así, si el universo consta de los números enteros no negativos menores que 10:

$$U=\{0,1,2,3,4,5,6,7,8,9\}$$

entonces podemos definir algunos conjuntos como:

$$A=\{0,2,4,6,8\}$$
$$B=\{1,3,5,7,9\}$$
$$C=\{1,4,7\}$$
Etcétera.

Con estas definiciones hemos establecido que cada uno de los elementos del universo pertenecen o no a un determinado conjunto. Por tanto, cada conjunto puede definirse completamente por una *función de pertenencia*, que opera sobre los elementos del universo, y que le asigna un valor de 1 si el elemento pertenece al conjunto, y de 0 si no pertenece.

Tomando como ejemplo el conjunto C enumerado arriba, su función de pertenencia $u_C(x)$ sería de la siguiente forma:

$$u_C(0)=0,\ u_C(1)=1,\ u_C(2)=0,\ u_C(3)=0,\ u_C(4)=1,\ u_C(5)=0,$$

$$u_C(6)=0,\ u_C(7)=1,\ u_C(8)=0,\ u_C(9)=0$$

Ahora bien, un *conjunto difuso* se define de forma similar, con una diferencia conceptual importante: un elemento puede pertenecer *parcialmente* a un conjunto. De esta forma, un conjunto difuso D definido sobre el mismo universo U puede ser el siguiente:

$$D=\{20\%/1,50\%/4,100\%/7\}^1$$

La definición anterior significa que el elemento *1* pertenece en un 20% al conjunto D (y por tanto pertenece en un 80% al complemento de D), en tanto que el elemento *4* pertenece en un 50%, y el elemento *7* en un 100% .

En forma alternativa, diríamos que la función de pertenecia $u_D(x)$ del conjunto D es la siguiente:

[1]. Se ha empleado una notación frecuente, en donde el signo no significa *dividido por*.

*Ingeniero Electricista, MSc.

$u_D(0)=0.0,\ u_D(1)=0.2,\ u_D(2)=0.0,\ u_D(3)=0.0,\ u_D(4)=0.5,$

$u_D(5)=0.0,\ u_D(6)=0.0,\ u_D(7)=1.0,\ u_D(8)=0.0,\ u_D(9)=0.0$

Las primeras diferencias que se hacen evidentes entre los conjuntos concretos y los conjuntos difusos son las siguientes:

- La función de pertenencia asociada a los conjuntos concretos sólo puede tener dos valores: 1 ó 0; mientras en los conjuntos difusos puede tener cualquier valor entre 0 y 1.

- Un elemento puede pertenecer (parcialmente) a un conjunto difuso y simultáneamente pertenecer (parcialmente) al complemento de ese conjunto. Lo anterior no es posible en los conjuntos concretos, ya que constituiría una violación al *principio del tercer excluido.*

- Las fronteras de un conjunto concreto son exactas, en tanto que las de un conjunto difuso son, precisamente, difusas, ya que existen elementos en las fronteras mismas, y estos elementos están a la vez dentro y fuera del conjunto.

¿Qué sentido puede tener pertenecer parcialmente a un conjunto? En muchos casos puede tener más sentido que pertenecer totalmente a un conjunto. Veamos algunos ejemplos:

Ejemplo 1: supóngase que se desea definir el conjunto de los estudiantes de la carrera de ingeniería eléctrica de la Universidad Nacional de Colombia que están cursando el quinto semestre de la carrera. ¿Cómo clasificar a un estudiante que cursa dos materias de cuarto semestre, tres de quinto y una de sexto? ¿Y a otro que toma una materia de quinto semestre, y cinco de sexto? Evidentemente ambos son *en parte* miembros del conjunto *Estudiantes de quinto semestre*, pero sólo parcialmente.

Ejemplo 2: supóngase que se desea clasificar a los miembros de un equipo de fútbol según su estatura en tres conjuntos, bajos, medianos y altos. Podría plantearse que se es bajo si se tiene una estatura inferior a, por ejemplo, 160 cm, que se es mediano si la estatura es superior o igual a 160 cm e inferior a 180 cm, y se es alto si la estatura es superior o igual a 180 cm, con lo que se lograría una clasificación en conjuntos concretos.

Sin embargo, ¿qué tan grande es la diferencia que existe entre dos jugadores del equipo, uno con estatura de 179,9 cm y otro de 180,0 cm? Ese milímetro de diferencia quizás no represente en la práctica algo significativo, y sin embargo los dos jugadores han quedado rotulados con etiquetas distintas: uno es mediano y el otro es alto. Si se optase por efectuar la misma clasificación con conjuntos difusos, estos cambios abruptos se evitarían, debido a que las fronteras entre los conjuntos permitirían cambios graduales en la clasificación.

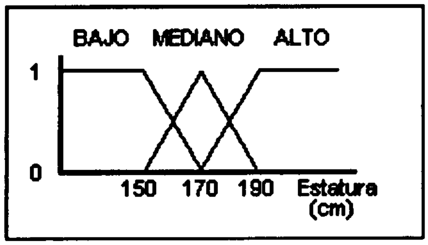

Figura 1.Funciones de pertenencia del ejemplo 2.

La figura 1 muestra cómo podría hacerse tal clasificación: el universo de discurso sería el conjunto continuo de todas las posibles estaturas (el intervalo [130 cm, 210 cm] por ejemplo). Las funciones de pertenencia de cada uno de los tres conjuntos bajo, mediano y alto se han graficado. La forma de estas funciones de pertenencia no debe ser necesariamente la de la figura 1, pues depende de lo que se entienda por *bajo, mediano* y *alto*. Las figuras 2 y 3 muestran otras alternativas para definir esas funciones.

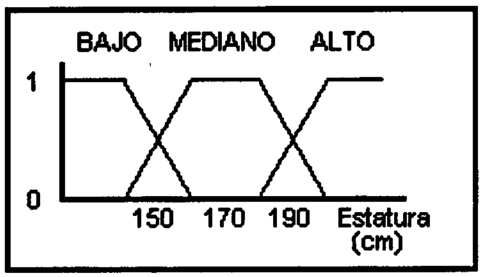

Figura 2. Representación alternativa del ejemplo 2.

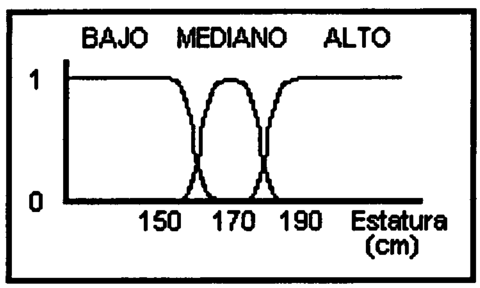

Figura 3. Representación alternativa del ejemplo 2.

II. Operaciones entre conjuntos difusos

Las tres operaciones básicas entre conjuntos concretos, unión, intersección y complemento, se definen también para los conjuntos difusos, intentando mantener el significado de tales operaciones. La definición de estas operaciones se hace empleando el concepto de función de pertenencia de los conjuntos.

Intersección: el resultado de efectuar la operación de intersección entre dos conjuntos difusos A y B definidos sobre el mismo universo, y con funciones de pertenencia $u_A(x)$ y $u_B(x)$, respectivamente, es un nuevo conjunto difuso ANB definido sobre el mismo universo, y con función de pertenencia $u_{ANB}(x)$, dada por:

$$u_{ANB}(x)= u_A(x)(*)u_B(x)$$

En donde el operador *(*)* debe satisfacer las siguientes propiedades:

$$x(*)y = y(*)x$$
$$(x(*)y)(*)z = x(*)(y(*)z)$$
$$\text{Si } x<y \text{ y } z<w, \text{ entonces } x(*)z < y(*)w$$
$$x(*)1=x$$

Todo operador que satisfaga las propiedades anteriores se conoce como una *T-norma*, y representa la intersección de dos conjuntos difusos[2]. Dos de los operadores más sencillos son el mínimo y el producto clásico (en adelante se denotarán por *mín.* y *, respectivamente). Las figuras 4 y 5 muestran la intersección de los conjuntos bajo y mediano de la figura 1, cuando se emplean los operadores mínimo y producto.

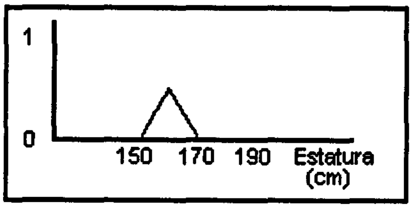

Figura 4.Operación intersección de los conjuntos bajo y mediano de la figura1 empleando el mínimo.

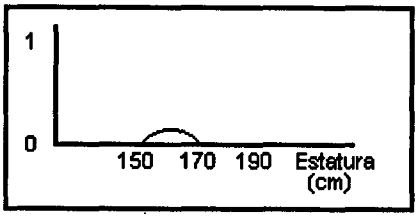

Figura 5.Operación intersección de los conjuntos bajo y mediano de la figura1 empleando el producto

Unión: el resultado de efectuar la operación de unión entre dos conjuntos difusos A y B definidos sobre el mismo universo, y con funciones de pertenencia $u_A(x)$ y $u_B(x)$, respectivamente es un nuevo conjunto difuso AUB definido sobre el mismo universo, y con función de pertenencia $u_{AUB}(x)$, dada por:

$$u_{AUB}(x)= u_A(x)(+)u_B(x)$$

En donde el operador *(+)* debe satisfacer las siguientes propiedades:

$$x(+)y = y(+)x$$
$$(x(+)y)(+)z = x(+)(y(+)z)$$
$$\text{Si } x<y \text{ y } z<w, \text{ entonces } x(+)z < y(+)w$$
$$x(+)0=x$$

Todo operador que satisfaga las propiedades anteriores se conoce como una *S-norma*, y representa la unión de dos conjuntos difusos[3]. Uno de los operadores más sencillo es el máximo (en adelante se denotará por *max.*). La figura 6 muestra la unión de los conjuntos bajo y mediano de la figura 1, cuando se emplea el operador máximo.

Complemento: el resultado de efectuar la operación de Complemento sobre un confunto difuso A definido sobre un Universo, y con funcion de pertenencia $u_A(x)$ es un nuevo conjunto difuso A' definido sobre el mismo universo, y con función de pertenencia $u_{A'}(x)$, dada por:

$$u_{A'}(x)= 1-u_A(x)$$

La figura 7 muestra el complemento del conjunto bajo de la figura 1.

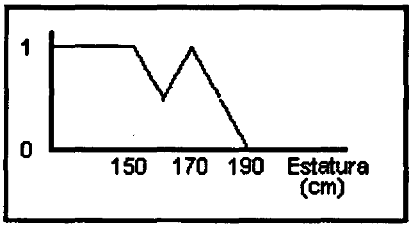

Figura 6. Operación unión de los conjuntos bajo y mediano de la figura 1 empleando el máximo.

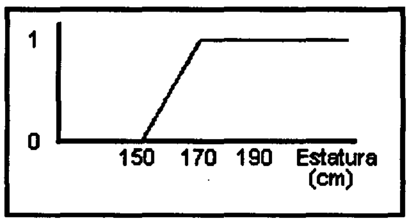

Figura 7.Operación complemento del conjunto bajo de la figura1

Otras operaciones como las relaciones entre conjuntos difusos, la composición de relaciones y el principio de extensión no se incluyen en este artículo. Se recomiendan [4], [6], [16] y [25].

[2]. Para una relación detallada de operadores *véase* [6].
[3]. Para una relación detallada de operadores *véase* [6].

III. Principios de lógica difusa

Es bien conocido que la teoría de conjuntos, el álgebra booleana y la lógica tradicional son isomorfas, bajo transformaciones adecuadas. Esto significa que tienen una estructura subyacente similar, y que por tanto las definiciones que se hagan en una cualquiera de las tres teorías se puede llevar a las otras dos, mediante transformaciones adecuadas. El cuadro 1 muestra la correspondencia de algunos operadores.

Cuadro 1. Correspondencia entre operadores de la teoría de conjuntos, el algebra booleana y la lógica tradicional.

Teoría de conjuntos	Álgebra booleana	Lógica tradicional
Intersección	Conjunción	AND
Unión	Disyunción	OR
Complemento	Negación	NOT

Ahora bien, el razonamiento lógico consiste en la combinación de proposiciones para producir nuevas proposiciones; así, la combinación de las proposiciones *«X es A»* y *«Y es B»* mediante el operador AND da como resultado la proposición *«X es A AND Y es B»*. El cuadro 1 sugiere que puede representarse esta combinación mediante un operador análogo a la intersección de conjuntos.

Lo anterior es posible porque en la lógica tradicional toda proposición puede tener uno de dos valores, *verdadero* o *falso*, lo que corresponde en la teoría de conjuntos concretos a los únicos dos valores que puede tomar la función de pertenencia para cualquier conjunto: *1* ó *0*.

Ahora bien, en lógica difusa una proposición puede representarse por un conjunto difuso: *«X es A»* corresponde a un conjunto *A* con función de pertenencia $u_A(x)$, mientras *«Y es B»* corresponde a un conjunto *B* con función de pertenencia $u_B(y)$, y la combinación de estas dos proposiciones con el operador AND, es decir, la proposición «*X es* A AND Y es B» corresponde a un nuevo conjunto difuso $A_{AND}B$ con función de pertenencia

$$u_{AANDB}(x,y)= min(u_A(x),u_B(y))$$

En donde se ha utilizado el operador mín. para efectuar la intersección de los dos conjuntos, pero en general podría haberse utilizado cualquier T-norma.

Nótese que los universos de discurso sobre los cuales están definidos los conjuntos A y B no son necesariamente el mismo; son, por ejemplo U y V respectivamente, mientras el conjunto $A_{AND}B$ está definido sobre el universo *UxV* (véase figura 8).

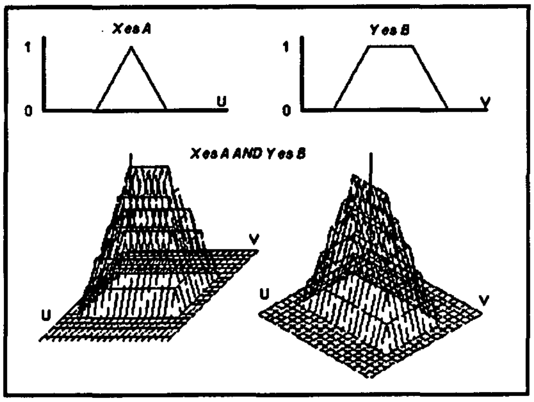

Figura 8. Operación AND

En forma análoga, al operador lógico *OR* puede hacerse corresponder a una *S-norma*, mientras al operador lógico *NOT* puede hacerse corresponder el complemento.

IV. Operador de implicación

Un análisis especial debe hacerse con el operador lógico de implicación =>, que combina dos proposiciones con la expresión *SI... ENTONCES... (IF ... THEN...)*, y que es el fundamento de las inferencias realizadas en sistemas de lógica difusa.

Ante todo, conviene precisar que el interés por el operador => consiste en encontrar una forma de interpretar proposiciones semejantes a las utilizadas en la experiencia común para describir conocimientos. Es decir, encontrar un camino matemático para evaluar proposiciones como las siguientes: *«Si las vibraciones son altas entonces el rodamiento está desgastado»*, o *«si los ingresos del cliente son bajos entonces su capacidad de endeudamiento es poca»*.

Ahora bien, la implicación => de la lógica tradicional tiene una tabla de verdad que se muestra en El cuadro 2.

Cuadro 2. Tabla de verdad de la implicación lógica tradicional.

p	q	p=>q
Verdad	Verdad	Verdad
Verdad	Falso	Falso
Falso	Verdad	Verdad
Falso	Falso	Verdad

Esta tabla de verdad puede obtenerse también con los operadores básicos conjunción, disyunción y negación con, por lo menos, dos expresiones distintas:

(p=>q)<=>(~(p^(~q)))
(p=>q)<=>((~p)óq)

Las anteriores equivalencias permiten deducir expresiones para la implicación de la lógica difusa. Para combinar dos proposiciones «X es A» y «Y es B» en la forma *«IF X es A THEN Y es B»*, debe representarse a cada una de dichas proposiciones por conjuntos difusos con funciones de pertenencia $u_A(x)$ y $u_B(y)$, respectivamente, y entonces la proposición combinada estará representada por un conjunto difuso $A=>B$, cuya función de pertenencia estará dada por:

$u_{A=>B}(x,y) = 1 - min(u_A(x),(1 - u_B(y)))$; o bien
$u_{A=>B}(x,y) = max(1 - u_A(x),u_B(y))$

No obstante, las expresiones anteriores (que llamaremos *implicaciones lógicas* o *implicaciones IF-THEN*) no son necesariamente las más útiles para efectuar inferencias, particularmente en aplicaciones de ingeniería. La razón puede hallarse revisando el cuadro 2: la implicación de la lógica tradicional es verdadera en tres condiciones, y sólo es falsa si la primera proposición es verdadera y la segunda es falsa, lo que puede interpretarse con la máxima: «La verdad nunca implica falsedad».

La tabla de verdad de la implicación indica en qué condiciones un razonamiento es formalmente correcto, pero no necesariamente útil. Veamos unos ejemplos:

«Si 1=2 entonces 3=3» es una implicación formalmente correcta, porque una falsedad (*1=2*) puede implicar una verdad (*3=3*), y para ello basta sumar al lado izquierdo *2* y al lado derecho *1* (recuerdese que partimos de *1=2*)[4].

De igual forma, la proposición *«Si 1=2 entonces 2=3»* tambien es formalmente correcta, porque una falsedad puede implicar una falsedad, y para ello basta sumar *1* a cada lado de la igualdad.

Los dos ejemplos anteriores son formalmente correctos, ¿pero qué utilidad puede extraerse de ellos en aplicaciones de ingeniería? En realidad sólo implicaciones en las que ambas proposiciones sean verdaderas pueden tener utilidad práctica, y esto es así porque las relaciones causa efecto son las que interesan en ingeniería, y no el formalismo de una implicación [16].

Visto lo anterior se concluye que las expresiones de implicación, útiles para efectuar inferencias lógicas, son en realidad operadores AND, es decir, *T-normas*. Al utilizar *T-normas* como implicaciones, llamamos a éstas *implicaciones de ingeniería* o *implicaciones AND*. Nuevamente, las *T-normas* más usadas como implicación son el mínimo y el producto.

Las figuras 9 y 10 muestran gráficamente la diferencia que existe entre emplear implicaciones lógicas e implicaciones de ingeniería. Se ha supuesto una expresión *«IF X es A THEN Y es B»*, en donde las proposiciones *«X es A»* y *«Y es B»* se han representado por los conjuntos que se muestran en las figuras. Nótese que la implicación de lógica llega a tener funciones de pertenencia *1* en zonas en donde los conjunto originales tienen funciones de pertenencia *0* (la falsedad puede implicar falsedad), en tanto que la implicación de ingeniería no lo hace así.

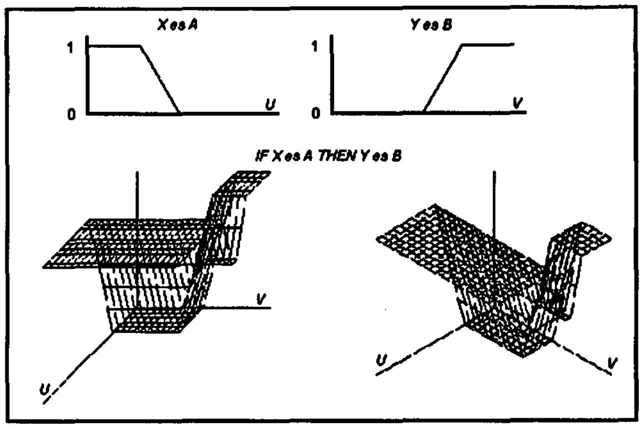

Figura 9. Ejemplo de una implicación IF-THEN o lógica.

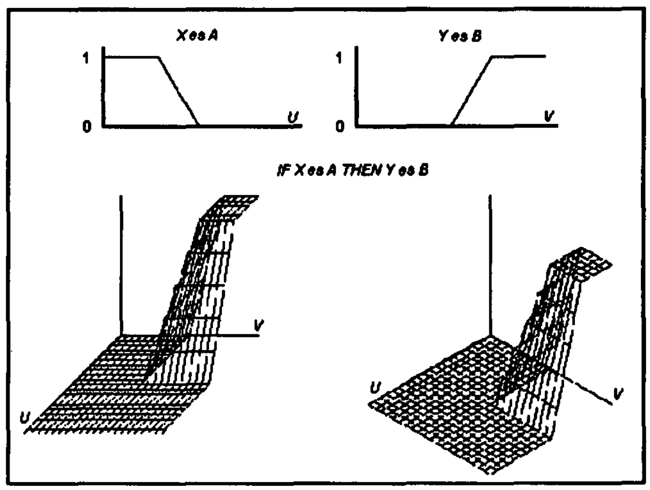

Figura 10. Ejemplo de una implicación AND o de ingeniería.

V. Inferencia en lógica difusa

La inferencia lógica consiste en la combinación de proposiciones para producir nuevas proposiciones. Así, al combinar la proposición *«X es A»* con la proposición *«IF X es A THEN Y es B»*, se puede inferir la proposición *«Y es B»* (véase figura 11)[5].

Una inferencia como la presentada en el párrafo anterior sólo es posible en la lógica tradicional si la primera proposición (*«X es A»*) es idéntica a la primera parte de la segunda proposición («*(IF) X es A*»); sin embargo, en la lógica difusa estas dos proposiciones no necesariamente deben ser idénticas, debido a que las fronteras de los conjuntos no son precisas.

[4]. Ejemplo tomado de [16]

[5]. Se ha presentado aquí un caso de aplicación de la regla de inferencia conocida como *modus ponens*, representado por $(p \wedge (p=>q))=>q$.

Así, al combinar la proposición *«X es A*»* con la proposición *«IF X es A THEN Y es B»*, puede obtenerse la proposición *«Y es B*»* (véase figura 12).

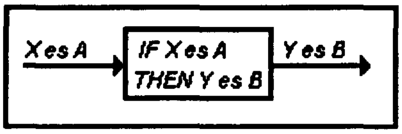

Figura 11.Inferencia en lógica tradicional.

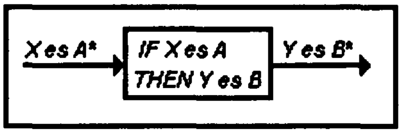

Figura 12. Inferencia en lógica difusa.

La combinación de estas proposiciones para efectuar la inferencia tiene su soporte matemático en la *extensión cilíndrica* y en la *composición de relaciones*, temas que no se han tratado en este artículo; sin embargo, la figura 13 muestra gráficamente cómo puede interpretarse esta inferencia.

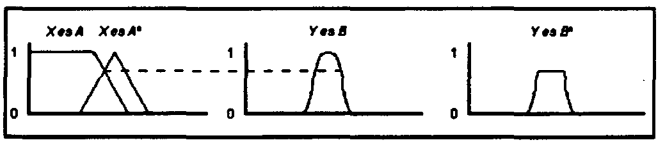

Figura 13. Representación gráfica de los mecanismos de inferencia en lógica difusa.

VI. Sistemas de lógica difusa

Los mecanismos de inferencia presentados en el numeral anterior permiten obtener conjuntos difusos a partir de la combinación de conjuntos difusos con reglas de la forma *IF... THEN...* Un sistema de lógica difusa aprovecha esos mecanismos como el motor de cálculo de un sistema cuyas entradas y salidas son números concretos.

La estructura básica de un sistema de lógica difusa se muestra en la figura 14. El sistema recibe varias entradas numéricas y entrega varias salidas numéricas. El bloque *difusor* se encarga de convertir las entradas en conjuntos difusos, que son entregados al bloque *máquina de inferencia*; este bloque, apoyado en un conjunto de reglas de la forma *IF... THEN...* almacenadas en la *base de reglas*, produce varios conjuntos difusos para que el bloque *concresor* los tome y los convierta en salidas numéricas concretas.

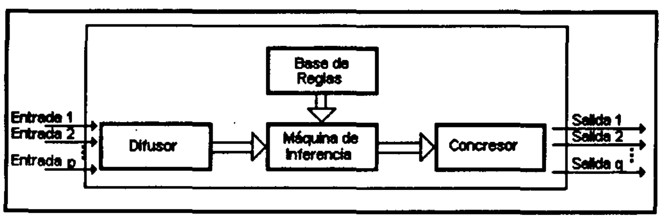

Figura 14. Estructura de un sistema de lógica difusa.

Cada una de las variables de entrada y de salida tiene una representación dentro del sistema de lógica lifusa en forma de *variables lingüísticas*. Una variable lingüística tiene, entre otras cosas, una colección de atributos que puede adquirir la variable, y cada atributo está representado por un conjunto difuso. Así, retomando el ejemplo de la figura 1, la variable *estatura* tendría tres atributos, *bajo, mediano y alto*, y cada uno de estos atributos estaría representado por el conjunto difuso respectivo de la figura 1. Estos atributos reciben el nombre de *valores lingüísticos*.

Debido a que un sistema de lógica difusa puede, en general, tener varias entradas y varias salidas, la forma genérica de las reglas presentes en la base de reglas es la siguiente:

IF X_1 *es* A_1 *AND* X_2 *es* A_2 *AND ... AND* X_m *es* A_m *THEN* Y_1 *es* B_1*AND* Y_2 *es* B_2 *AND... AND* Y_n *es* B_n

En estas reglas, A_1,A_2,..., A_m,B_1 ,B_2,...,B_n son valores lingüísticos de las variables lingüísticas respectivas.

El siguiente ejemplo sencillo quizás ayude a entender la estructura de un sistema de lógica difusa:

Ejemplo 3: una entidad financiera necesita determinar cuánto dinero puede prestarle a sus clientes. Para ello quiere utilizar como únicos criterios de evaluación los ingresos mensuales y el promedio de ahorro mensual de cada cliente. Se propone como solución un sistema de lógica difusa con las siguientes características:

Figura 15. Sistema del ejemplo 3.

El sistema recibe dos entradas, el ingreso mensual y el promedio mensual de ahorro y entrega una salida, el monto máximo del préstamo (véase figura 15). Estas tres variables se representan internamente por las variables lingüísticas ingreso, ahorro y préstamo, cuyos valores lingüísticos se muestran en la figura 16 [6], y se han consignado en el cuadro 3.

Cuadro 3 . Valores lingüísticos del ejemplo 3.

Ingreso	Ahorro	Préstamo
Muy bajo	Bajo	Muy pequeño
Bajo	Medio	Pequeño
Medio	Alto	Poco pequeño
Alto		Normal
Muy alto		Poco grande
		Grande
		Muy grande

[6].La escala de las tres gráficas se ha normalizado al intervalo [0,1].

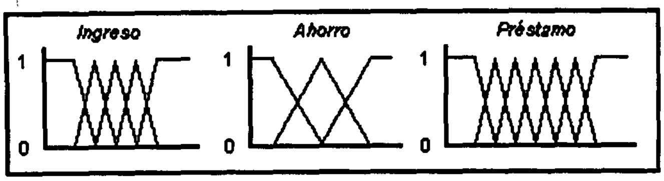

Figura 16 . Valores lingüísticos del ejemplo 3.

Las reglas que deben existir en la base pueden obtenerse con un poco de sentido común; por ejemplo, si el ingreso es muy bajo y el ahorro es bajo, el préstamo debe ser muy pequeño, mientras que si el ingreso es muy alto y el ahorro es alto, el préstamo debe ser muy grande. Lo anterior significa que deben existir por lo menos las dos reglas siguientes:

IF ingreso es muy bajo AND ahorro es bajo THEN préstamo es muy pequeño.

IF ingreso es muy alto AND ahorro es alto THEN préstamo es muy grande.

En forma similar pueden obtenerse las demás reglas, que se presentan resumidas en el cuadro 4.

Cuadro 4 . Reglas del ejemplo 3.

AHORRO	INGRESOS				
	Muy bajo	Bajo	Medio	Alto	Muy alto
Bajo	Muy pequeño	Pequeño	Poco pequeño	Normal	Poco grande
Medio	Pequeño	Poco pequeño	Normal	Poco grande	Grande
Alto	Poco pequeño	Normal	Poco grande	Grande	Muy grande

Nótese que el diseño de las variables lingüísticas y de la base de reglas ha seguido criterios subjetivos, pero extraídos del sentido común, y no ha sido necesario plantear complejos modelos matemáticos. Aun así, el sistema diseñado permite solucionar el problema planteado, con algunas características interesantes; por ejemplo, si las políticas crediticias de la entidad cambian para restringir los préstamos, basta modificar algunas casillas del cuadro 4 para adecuar el sistema, o bien se pueden modificar las funciones de pertenencia de la figura 16.

Ahora bien, el diseñador también debe seleccionar varias opciones matemáticas dentro del sistema de lógica difusa. Este punto no se trata en el presente artículo, por ser tan sólo una introducción al tema, y se remite al lector a [4], [5] y [6].

La figura 17 muestra los resultados producidos por el sistema del ejemplo 3 con algunas de las opciones matemáticas más utilizadas. Se ha graficado el monto máximo del préstamo en función del ingreso mensual, tres condiciones distintas de ahorro medio mensual.

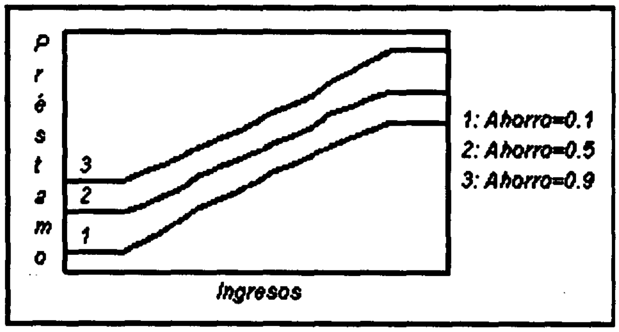

Figura 17. Resultados del sistema del ejemplo 3.

VII. Entrenamiento de Sistemas de Lógica Difusa[7]

Cuando un sistema de lógica difusa cuenta con un mecanismo de entrenamiento, se dice que es un *sistema difuso adaptativo*[8]. Los mecanismos de entrenamiento son algoritmos que le permiten al sistema cambiar su diseño para ajustarse (esto es, para adaptarse) a algunas exigencias específicas.

En general, los algoritmos de entrenamiento diseñan sólo una parte del sistema de lógica difusa, generalmente la base de reglas, o la definición de las variables lingüísticas, o en algunos casos ambas cosas. Los demás parámetros los debe seleccionar el usuario.

A nivel mundial, éste es uno de los temas sobre el que más se investiga actualmente dentro de los tópicos de lógica difusa. Existen diversos algoritmos, y distintas estrategias dependiendo de la utilización que esté dándosele al sistema de lógica difusa. Pero, ¿qué justifica este esfuerzo a nivel mundial? Existen por lo menos las siguientes razones:

En primer lugar, algunos sistemas de lógica difusa[9] son *aproximadores universales*, es decir, satisfacen una propiedad según la cual se sabe que cualquier función real continua puede ser aproximada con el grado de precisión que se desee por uno de estos aproximadores.

Esta propiedad asegura entonces la existencia de un sistema de lógica difusa con el que puede representarse, tan bien como se quiera, cualquier función no lineal continua. Sin embargo, aunque se sabe que tal sistema existe, no se conoce un procedimiento exacto para saber cuál es. En general, los algoritmos de entrenamiento son procedimientos lógicos que intentan diseñar un sistema de lógica difusa que aproxime alguna función desconocida.

---

7. Debido al alcance que se la ha dado a este artículo, la siguiente es una presentación de los mecanismos de entrenamiento de sistemas de lógica difusa *en general*, es decir, sin entrar a detallar ninguno de los mecanismos conocidos en la literatura técnica. Para una presentación detallada, véanse [25],[27],[28],[29].

8. *Véase* [25] página 2.

9. Por ejemplo, aquellos con difusor singleton, inferencia y operaciones AND con producto y concresor de altura.

En segundo lugar, un sistema como el del ejemplo 3 está basado principalmente en el conocimiento (expresado lingüísticamente) que se tiene sobre un cierto problema, en este caso la asignación de crédito. Sin embargo, en muchas ocasiones este conocimiento es insuficiente, o se encuentra acompañado de información numérica. Tal es el caso de muchas plantas industriales, donde además de un conocimiento general sobre el comportamiento de la planta, pueden existir registradores que midan y almacenen algunas de las variables del proceso.

Los algoritmos de entrenamiento son capaces de incorporar esta información numérica, junto con la información lingüística en un mismo sistema de lógica difusa. Esta unión de los dos tipos de conocimiento, lingüístico y numérico, en un mismo marco conceptual, hace de los sistemas difusos adaptativos algo excepcional.

Un sistema difuso adaptativo puede entonces intentar diseñarse él mismo para cumplir una función específica. Esta propiedad de *autoorganización* hace que sea sensato proponer los sistemas de lógica difusa como solución a problemas complejos, en los que las representaciones matemáticas exactas no se conocen, o son lo suficientemente complicadas como para que no sea práctico emplearlas.

Tal como se afirma unos párrafos arriba, la investigación en algoritmos de entrenamiento es uno de los temas de mayor auge en la actualidad, y por esa razón aún es muy temprano para poder sopesar adecuadamente las bondades de uno u otro algoritmo en aplicaciones reales. No obstante, puede hacerse una distinción entre dos tipos de algoritmos, según el efecto final que tienen sobre el sistema de lógica difusa.

Nótese que un sistema como el del ejemplo 3 es fácil de entender para una persona que no haya participado en el diseño del mismo, porque la manera en que ha quedado expresado el conocimiento en forma de reglas es clara, y la definición de los valores lingüísticos corresponde tambien a conceptos sencillos. En otras palabras, a partir del sistema del ejemplo 3 una persona puede extraer conocimiento de tipo lingüístico sobre el proceso de asignación de crédito, tan sólo observando la figura 16 y el cuadro 4.

Pues bien, algunos algoritmos de entrenamiento diseñan el sistema en forma tal, que una vez concluido el diseño es virtualmente imposible entenderlo, es decir, es imposible extraer conocimiento lingüístico del sistema diseñado. Otros, por su parte, efectúan un diseño tal que es posible emplearlo para interpretar el sistema diseñado. Los primeros son algoritmos *aproximativos* y los segundos son algoritmos *descriptivos*.

VIII. UNFUZZY: *SOFTWARE* DE LÓGICA DIFUSA

En el Departamento de Ingeniería Eléctrica de la Universidad Nacional de Colombia se ha elaborado una herramienta para al análisis, diseño, simulación e implementación de sistemas de lógica difusa; este *software* se ha denominado UNFUZZY[10], y está disponible en internet[11]. Las principales características de esta herramienta son las siguientes:

- Permite el diseño gráfico de los universos de entrada y salida.
- Permite seleccionar algoritmos de difusión, concreción, implicación, composición, unión-intersección y operadores AND, dentro de un conjunto amplio de opciones.
- Permite el diseño de bases de reglas completas o incompletas, así como la opción de incluir modificadores lingüísticos.
- Provee herramientas de diseño rápido para los universos de entrada y salida, las variables lingüísticas y la base de reglas.
- Permite analizar el comportamiento global del sistema mediante gráficos y tablas entrada-salida.
- Permite analizar el comportamiento del sistema a entradas particulares mediante la presentación paso a paso de los resultados intermedios de cada algoritmo.
- Permite entrenar sistemas de lógica difusa mediante el uso de tablas, a través de dos algoritmos diferentes.
- Provee el código fuente C y C++ del sistema diseñado.

El objetivo inicial del proyecto era el diseño e implementación de un *software* que permitiera diseñar sistemas de lógica difusa, y analizar su comportamiento. UNFUZZY cumple tales objetivos, y además los supera, ya que incluye dos funciones adicionales que le dan un potencial mucho mayor:

- La generación de código fuente en lenguaje C y C++.
- La opción de entrenamiento de sistemas de lógica difusa mediante tablas.

La posibilidad de disponer de un código fuente probado le permite al usuario implementar en *software* el sistema de lógica difusa diseñado. En otras palabras, no sólo puede analizar el comportamiento del sistema, sino que además puede disponer de él para su utilización particular, sin tener que preocuparse por cuáles son los algoritmos internos del sistema. El usuario sólo debe utilizar una herramienta gráfica de diseño, y ésta pone en sus manos el código fuente.

---

[10]. Software diseñado por el autor como tesis de maestría en automatización industrial. Véase [5].

[11].http://ohm.ingsala.unal.edu.co/ogduarte. seleccionar *software*.

La opción de entrenar sistemas de lógica difusa amplía el tipo de aplicaciones en los que éstos son utilizables. Vale la pena resaltar que el código C++ que genera UNFUZZY incluye la opción de entrenamiento, y por tanto el usuario dispone de ella en su aplicación particular.

AGRADECIMIENTOS

Debo especial agradecimiento al ingeniero y amigo Gustavo Pérez Hoyos Ph.D. por su labor en la dirección de la tesis de maestría en automatización industrial, cuyo resultado fue el programa UNFUZZY. El ingeniero Alberto Delgado Ph.D. suministró la bibliografia en la que aparece como autor o coautor Kevin Passino.

BIBLIOGRAFÍA

1. ÁLVAREZ, Hernán D. "Control difuso y sistemas de control inteligentes", en: *Memorias del Segundo Congreso de la Asociación Colombiana de Automática*. Bucaramanga, Colombia, marzo de 1997. pp. 331-340.
2. ANGASANA, A. y PASSINO, K. "Distributed fuzzy control of flexible manufacturing systems". *IEEE Transactions on Control Systems Technology*. Vol. 2, No.4, dic. 1994. pp. 423-435.
3. ANTASKLIS, P.J. y PASSINO, K. "Introduction to intelligent control with high degrees of autonomy", en: *An Introduction to Intelligent Control and Autonomous Systems*. Kluwer Academic Publishers, 1993. pp. 1-26.
4. DRIANKOV, Dimiter *et al. An Introduction to Fuzzy Control*. Springer Verlag, Berlin. 1993.
5. DUARTE, Oscar G. UNFUZZY. *Software para el análisis, diseño, simulación e implementación de sistemas de lógica difusa*. Tesis de Magister. Universidad Nacional de Colombia, Facultad de Ingeniería, Maestría en Automatización Industrial, Santa Fe de Bogotá. 1997.
6. KLIR, George & YUAN, Bo. *Fuzzy Sets and Fuzzy Logic*. Prentice Hall, New Jersey. 1995.
7. KWONG, W. *et al.* "Expert supervision of fuzzy learning systems for fault tolerant aircraft control". *Proceedings of the IEEE*. Vol. 83, No. 3, mar. 1995. pp 466-483.
8. KWONG, W. y PASSINO, K. "Dynamically focused fuzzy learning control". *IEEE Transactions on Systems, Man and Cybernetics - Part B :Cybernetics*. Vol. 26, No.1, febrero 1996. pp 53-74.
9. LAUKONEN, E. y PASSINO, K. "Training fuzzy systems to perform estimation and identification". *Engng. Appl. Artif. Intel.* Vol 8, No. 5, Elsevier Science Ltd. Gran Bretaña, pp. 449-514.
10. LAUKONEN, E. *et al.* "Fault detection and isolation for an experimental internal combustion engine via fuzzy identification". *IEEE Transactions on Control Systems Technology*. Vol. 3, No. 3, septiembre 1995. pp. 347-355.
11. LAYNE, J. y PASSINO, K. "Fuzzy model reference learning control for cargo ship steering". *IEEE Control Systems Magazine*. Vol. 13, No. 6, diciembre 1993. pp. 23-24.
12. LEE, Chuen Chien. "Fuzzy logic in control systems: fuzzy logic controller-part I". *IEEE Transactions on Systems, Man, and Cibernetics*. Vol. 20, No. 3, marzo/abril 1990. pp. 404-418.
13. __________ "Fuzzy logic in control systems: fuzzy logic controller-part II". *IEEE Transactions on Systems, Man, and Cibernetics*. Vol. 20, No. 3, marzo/abril 1990. pp. 419-435.
14. LIN, Chin-Teng y LEE, George. *Neural Fuzzy Systems. A NeuroFuzzy Synergism to Intelligent Systems*. Prentice Hall, 1996.
15. LÓPEZ, José F. *et al.* "Comparación entre sistemas difusos para el control de una banda transportadora dentro de un proceso de manufactura flexible" en: *Memorias del Segundo Congreso de la Asociación Colombiana de Automática*. Bucaramanga, Colombia, marzo de 1997. pp. 280-288.
16. MENDEL, Jerry. "Fuzzy logic systems for engineering: a tutorial". *Procceedingos of the IEEE*. Vol. 83, No. 3, marzo 1995. pp. 345-377.
17. MENESES, Jorge E. *et al.* "Control fuzzy basado en microcontrolador aplicado a la operación de un válvula remota", en: *Memorias del Segundo Congreso de la Asociación Colombiana de Automática*. Bucaramanga, Colombia, marzo de 1997. pp. 175-181.
18. MOUDGAL, V yPASSINO, K. "Rule-based control for a flexible-link robot". *IEEE Transactions on Control Systems Technology*. Vol. 2, No. 4, diciembre 1994. pp. 392-405.
19. ORDÓÑEZ, R. *et al.* "Stable multiple-input multiple-output adaptative fuzzy control", en: *Proceedings of the 35th Conference on Decision and Control*. Kobe, Japan, diciembre 1996. pp. 610-615.
20. PASSINO, Kevin. "Intelligent control". *The Control Handbook, IEEE*. Pp. 994-1001.
21. __________. "Fuzzy control". *The Control Handbook*, IEEE. Pp. 1001-1017.
22. __________. "Intelligent control for autonomous systems". *IEEE Spectrum*. Junio 1995. pp. 55-62.
23. PASSINO, K. y LUNARDHI, A. "Qualitative analysis of expert control systems". *Intelligent Control Systems : Theory and Applications*. IEEE Press, N.Y. 1996.
24. SOLANO, Karim *et al.* "Controladores adaptables basados en mecanismos de inferencia difusa", en: *Memorias del Segundo Congreso de la Asociación Colombiana de Automática*. Bucaramanga, Colombia, marzo de 1997. pp. 104-113.
25. WANG, Li-Xin. *Adaptative Fuzzy Systems and Control. Design and Stability Analysis*. Prentice Hall, New Jersey. 1994.
26. __________ "Fuzzy systems are universal aproximator". *Proceedings of the IEEE International Conference on Fuzzy Systems*. San Diego, California, 1992. pp. 1163-1170.
27. __________ & MENDEL, Jerry. "Back propagation fuzzy system as nonlinear dynamic systems identifiers". *Proceedings of the IEEE International Conference on Fuzzy Systems*. San Diego, California, 1992. pp. 1409-1418.
28. __________, Jerry. "Fuzzy basics functions, universal approximation, ang orthogonal least-squares learning". *IEEE Transactions on Neural Networks*. Vol. 3, No. 5, septiembre 1992. pp. 807-814.
29. __________, "Generating fuzzy rules by learning from examples". *Proceedings of the IEEE 1991 International Symposium on Intelligent Control*. August 13-15, Arlington, Virginia, U.S.A. pp. 263-268.
30. YAMAKAWA, T. "A fuzzy inference engine in nonlinear analog mode and its applications to a fuzzy logic control". *IEEE Transactions on Neural Networks*. Mayo de 1993.