VNIVERSIDAD NACIONAL
AVFNOMA DE
MEXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

"SISTEMAS ADAPTABLES BASADOS EN LA LÓGICA DIFUSA Y SU APLICACIÓN EN EL CONTROL DE NIVEL DE LÍQUIDO"

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRA EN CIENCIAS (COMPUTACIÓN)

P R E S E N T A:

GUO HUA SUN WU

DIRECTOR DE TESIS: DR. YU TANG XU

México, D.F. 2005.

A mis profesores.

A Ray, Paty, Javier, Ricardo, German y a todos mis compañeros de la maestría.

A Marcos, Antonio, Carlos, Miriam y a todos mis amigos de la UNAM.

A Lulú, Amalia, Diana y Juanita.

GRACIAS POR SU APOYO Y AMISTAD

Sistemas Adaptables Basados en la Lógica Difusa y su Aplicación en el Control de Nivel de Líquido

Guo-Hua Sun

Posgrado en Ciencia e Ingeniería de la Computación

Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas

Mayo de 2005

Índice general

Índice de figuras

Capítulo 1

Introducción

En este capítulo, se presentan antecedentes del conocimiento sobre este trabajo. La motivación y la meta de la investigación son presentadas en la sección 1.1, una explicación breve de los temas relativos en la literatura está en la sección 1.2 y 1.3, la sección 1.4 es el resumen de todo el trabajo de tesis.

1.1. Motivación y objetivos

La teoría de conjuntos difusos fue introducida por Zadeh en 1965, de allí viene la derivación del nombre lógica difusa. Desde su primera aplicación exitosa en la ingeniería de control en 1974 [26], la lógica difusa se ha expandido en todas las ramas de la ingeniería.

Los sistemas basados en la lógica difusa usan reglas lingüísticas para describir sistemas, por lo tanto permiten representar y manipular información en forma similar a los procesos de razonamiento y comunicación humanos. Estos sistemas son convenientes para manejar problemas complejos donde es muy difícil, o inclusive imposible, describirlos con modelos matemáticos convencionales.

En el diseño de sistemas difusos, una tarea importante es la generación de las reglas difusas. En muchas aplicaciones, las reglas difusas son generadas por expertos del área. Sin embargo, con un número creciente de variables, el número posible de reglas para el

sistema aumenta exponencialmente, y es difícil definir un conjunto de reglas completo con un buen desempeño. Así un método automático de diseñar sistemas difusos es necesario.

El diseño de un sistema difuso puede ser formulado como un problema de búsqueda en un espacio de gran dimensión en donde cada punto representa un conjunto de reglas, funciones de membresía y el comportamiento correspondiente del sistema. Lograr el diseño de un sistema difuso óptimo es equivalente a buscar un punto óptimo en este espacio para que el comportamiento sea el deseado.

Desde este punto de vista, los algoritmos genéticos, técnica inspirada por el proceso natural de evolución de las especies, puede ser un buen candidato para resolver este problema, dada la capacidad de los algoritmos genéticos para buscar una solución óptima en un espacio de búsqueda de alta dimensión y/o muy complejo.

El objetivo principal de esta tesis es, entonces, investigar una metodología para el diseño de sistemas adaptables basado en la lógica difusa y algoritmos genéticos.

Con esta metodología definida se pretende controlar el nivel de líquido en un sistema de dos tanques, que es un problema importante en procesos industriales, el cual se observa ampliamente en sistemas de la industria química, petroquímica, de papel y de tratamiento de agua. La metodología para el diseño tiene como objetivo:

a) Obtener un diseño automático del sistema difuso.

b) Optimizar el diseño con respecto a un criterio dado.

c) Disminuir la intervención del experto humano en la definición de la estructura del sistema.

d) Disminuir el tiempo de diseño del sistema de control.

1.2. Lógica Difusa y Algoritmos Genéticos

La lógica difusa es una lógica de razonamiento aproximado, la cual puede ser vista como una generalización y extensión de la lógica multivaluada. A diferencia de las matemáticas y ciencias convencionales, la teoría difusa maneja los problemas en donde

las variables son difíciles de cuantificar y las dependencias no están bien definidas.

La lógica difusa ha cobrado una gran fama por la variedad de aplicaciones en las que puede ser usada, éstas van desde el control de complejos procesos industriales, hasta el diseño de dispositivos artificiales de deducción automática, pasando por la construcción de artefactos electrónicos de uso doméstico y de entretenimiento, así como también de sistemas de diagnóstico. De hecho, desde hace ya al menos década y media, la expedición de patentes industriales de mecanismos basados en la lógica difusa ha tenido un crecimiento muy rápido en todas las naciones industrializadas del orbe.

Los algoritmos genéticos(AGs) son algoritmos de búsqueda que simulan los procesos de selección y genética naturales, y tienen diversas aplicaciones en la física, la computación, las ciencias sociales y la ingeniería. Los AGs fueron introducidos por *John Holland* en 1975. Los procesos de búsqueda, basado en el principio de *Darwin* "los mejores sobreviven", permiten que las soluciones evolucionen a soluciones superiores. En el libro de *Holland* "Adaptation in natural and artificial systems"se presentan operadores genéticos, como los de selección, recombinación y mutación.

1.3. Sistemas Adaptables

Un sistema adaptable generalmente se refiere a un sistema que cambia su comportamiento respondiendo al ambiente en donde se está operando. El cambio que ocurre es siempre relevante a una meta o un objetivo.

Particularmente, un sistema adaptable basado en la lógica difusa, es un sistema equipado con un algoritmo de entrenamiento donde el sistema es construido por una colección de reglas difusas *Si-entonces*. Se podrían llevar a cabo en 3 niveles de adaptación [48]. En primer término está la adaptación de estructura, donde *French* y sus colaboradores [9] mencionaron que al seleccionar los conjuntos de variables manipuladas, la estructura de entrada y el tipo de sistema es modificado. Entonces el cambio adecuado de estos conjuntos puede ser considerado como un caso especial para incrementar el desempeño del sistema. Los otros dos niveles son la adaptación del algoritmo de apren-

dizaje y la adaptación de parámetros tales como los de funciones de membresía [29]. Con estas adaptaciones se logrará el cambio que se conduce el comportamiento del sistema sea más adecuado. Esta tesis, se concentrará en la adaptación de parámetros del sistema por ser una técnica efectiva y no tan compleja.

1.4. Organización de la tesis

En este trabajo se implantará un sistema adaptable a través de un sistema de lógica difusa. El sistema adaptable recibe señales del ambiente en donde está sumergido, mediante algoritmos de adaptación y de acuerdo con ciertos criterios de desempeño adapta su comportamiento para que el desempeño del sistema global sea el deseado. Por lo tanto, en la tesis primero se revisará los trabajos relevantes en la literatura, luego se investigará desde el punto de vista de sistemas adaptables el sistema difuso, así como los algoritmos de adaptación, para después desarrollar un nuevo diseño de sistemas adaptables basado en la lógica difusa y algoritmos genéticos. La aplicación de este diseño al control de nivel de líquido será presentada con detalle para ilustrar la metodología.

En esta tesis se hará una breve descripción de sistemas difusos, se desarrollará el sistema adaptable basado en los sistemas de lógica difusa y se llevará a cabo el diseño de un controlador basado en la lógica difusa utilizando el algoritmo genético propuesto para controlar el nivel de líquido.

En el capítulo 2 se presentan las ideas y conceptos básicos sobre sistemas de lógica difusa *(SLDs)*, así como una breve descripción de la teoría difusa, estructura de *SLDs* y diseño de un *SLD*.

Las descripciones de sistemas adaptables basados en lógica difusa, de los algoritmos de adaptación (los cuales incluyen los algoritmos de retropropagación, mínimos cuadrados ortogonal y los algoritmos genéticos propuestos), y un diseño automático de sistemas adaptables se detallan en el capítulo 3.

En el capítulo 4 se muestra la aplicación de la metodología propuesta al problema

de control del nivel de líquido en un sistema de dos tanques. Se presenta el diseño del controlador difuso adaptable y se incluyen los resultados de simulación.

Finalmente las conclusiones y observaciones de este trabajo se presentan en el capítulo 5.

Capítulo 2

Sistemas de Lógica Difusa (SLDs)

Un *SLD* es un marco de computación basado en los conceptos de la teoría de conjuntos difusos, reglas difusas de la forma *Si-entonces* y razonamiento difuso. En este capítulo se presentarán primero los conceptos básicos acerca de la lógica difusa.

2.1. Lógica difusa

En la vida real existen hechos inciertos o conceptos que albergan valores de verdad que se encuentran entre la pertenencia y la no-pertenencia. Por ejemplo: "¿ Es Juan alto? o "¿Es Maria inteligente? Para estos casos, la lógica matemática convencional no responde con eficacia. Eso da lugar a un nuevo tipo de lógica, llamada lógica difusa, la cual puede determinar ese tipo de valores intermedios, y está basada en el concepto de conjuntos difusos creado por *Zadeh.*

2.1.1. Conjuntos difusos

En 1965, un catedrático en la Universidad de California en Berkeley, *Lofti A. Zadeh* publicó un artículo " Fuzzy sets"[24]. De entonces a la fecha, el término *difuso* se relaciona con cualquier sistema matemático o computacional que tenga que ver con los conjuntos difusos. A partir del desarrollo de productos (tren o metro) en Japón

que utilizan estas herramientas o teoría, se le ha prestado más atención en el área de procesos e ingeniería de control.

Un conjunto difuso es un conjunto donde la pertenencia no tiene un límite preciso claramente definido. Supongamos que U es el universo de discurso, el cual contiene todos los posibles elementos en cualquier contexto particular. Un conjunto clásico (preciso) A se puede presentar usando el método de regla como:

$$A = \{x \in U \mid x \; satisface \; algunas \; condiciones\}$$

También podemos usar el método de membresía a definir el conjunto A, la función de membresía *0-1* para A está denotada por $\mu_A(x)$, tal como

$$\mu_A(x) = \begin{cases} 1, & sii \; x \in A, \\ 0, & sii \; x \notin A. \end{cases}$$

En este caso, el grado de membresía del conjunto preciso A es 1 o 0. Si el grado puede ser cualquier punto (gradualmente) en el intervalo $[0, 1]$, se denomina a A un *conjunto difuso*. Conjuntos difusos extienden el significado de un conjunto dando lugar a diferentes *grados de pertenencia*, también se llaman *valores de membresía*. Estos valores son definidos por una *función de membresía* que es una función continua que determina, para cada elemento del conjunto difuso, un valor de membresía μ, entre 0 y 1.

Generalmente se usan las funciones triangular, trapezoidal, gaussiana, sigmoides, etc. como funciones de membresía [34]. A continuación se muestran las funciones usadas frecuentemente y su representación gráfica en la figura 2.1.

Función de membresía triangular

$$f_{triangular}(x) = \begin{cases} 0, & \text{si } x < x_1, \\ 2\dfrac{x - x_1}{x_2 - x_1}, & \text{si } x_1 \leq x \leq \dfrac{x_1 + x_2}{2}, \\ 2\dfrac{x_2 - x}{x_2 - x_1}, & \text{si } \dfrac{x_1 + x_2}{2} \leq x \leq x_2, \\ 0, & \text{si } x > x_2. \end{cases} \tag{2.1}$$

Función de membresía triangular-izquierda

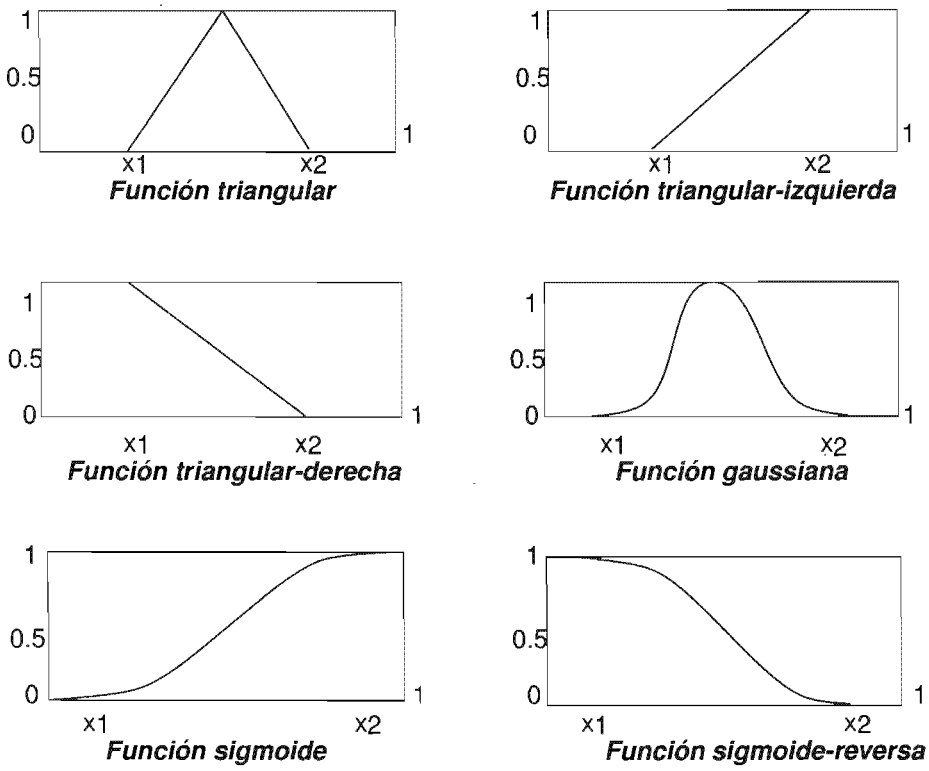

Figura 2.1: Algunas funciones de membresía.

$$f_{triangular-izquierda}(x) = \begin{cases} 0, & \text{si } x < x_1, \\ \dfrac{x_2 - x}{x_2 - x_1}, & \text{si } x_1 \leq x \leq x_2, \\ 1, & \text{si } x > x_2. \end{cases} \tag{2.2}$$

Función de membresía triangular-derecha

$$f_{triangular-derecha}(x) = \begin{cases} 1, & \text{si } x < x_1, \\ \dfrac{x - x_1}{x_2 - x_1}, & \text{si } x_1 \leq x \leq x_2, \\ 0, & \text{si } x > x_2. \end{cases} \tag{2.3}$$

Función de membresía gaussiana

$$f_{gaussiana}(x) = e^{-y^2/2}, \quad y = \frac{m(x - x_1)}{x_2 - x_1} - n, \quad m, n \text{ son constantes.} \tag{2.4}$$

Función de membresía sigmoide

$$f_{sigmoide}(x) = \frac{1}{1 + e^{-y}}, \quad y = \frac{m(x - x_1)}{x_2 - x_1}, \quad m \text{ es constante.} \tag{2.5}$$

Función membresía sigmoide-reversa

$$f_{sigmode-reversa}(x) = 1 - f_{sigmode}(x). \tag{2.6}$$

Así como en las conjuntos clásicos existen los operadores **Intersección, Unión y Complemento** [43], así mismo sucede con los conjuntos difusos.

Dados A y B dos conjuntos difusos en U,

La intersección $A \bigcap B$ de A y B es un conjunto difuso en U con la función de membresía definida para todo $u \in U$ por

$$\mu_{A \cap B}(u) = \text{mín}\{\mu_A(u), \mu_B(u)\}. \tag{2.7}$$

La unión $A \bigcup B$ de A y B es un conjunto difuso en U con la función de membresía definida para todo $u \in U$ por

$$\mu_{A \cup B}(u) = \text{máx}\{\mu_A(u), \mu_B(u)\}. \tag{2.8}$$

Normalmente los operadores de intersección y unión son denotados por $\wedge$ y $\vee$, respectivamente.

El *complemento* $\bar{A}$ de A es un conjunto difuso en U con la función de membresía definida para todo $u \in U$ por

$$\mu_{\bar{A}}(u) = 1 - \mu_A(u). \tag{2.9}$$

2.1.2. Reglas difusas

Para conocer reglas difusas, se debe conocer primero un componente básico: **la variable lingüística.**

Para cuantificar una variable, en matemáticas, digamos la velocidad X se asigne a ésta un número 80 km/hr, o una constante a, etc. Sin embargo en lógica difusa, una variable puede tomar palabras como sus valores. Por ejemplo, " la *velocidad* es *muy alta* ", aquí "*la velocidad*", es una variable lingüística.

En este caso, consideramos una definición intuitiva: Si una variable puede tomar palabras en lenguaje natural (por ejemplo, rápido, bajo, etc.) como sus valores, esta variable es definida como una variable lingüística. Estas palabras frecuentemente tienen que ver con conjuntos difusos. Una nota importante es que una variable lingüística puede tomar *palabras* o *números* como sus valores.

Teóricamente, cada variable difusa puede tener muchos conjuntos difusos mientras cada conjunto difuso tiene sus propias funciones de membresía y generalmente se usan tres, cinco, siete, o hasta nueve conjuntos difusos para cada variable difusa.

Teniendo en cuenta lo anterior para las **reglas difusas** *Si-entonces* **(declaraciones difusas condicionales)** [**37**] se tiene:

Una regla difusa es una regla del tipo

$$Si\ x\ es\ A,\quad entonces\ y\ es\ B.$$

en el cual A y B son etiquetas lingüísticas definidas en sus respectivos universos U y V. La parte $Si\ x\ es\ A$ es el antecedente y $y\ es\ B$ el consecuente. Por ejemplo: Si un durazno está rosa, entonces está maduro.

Por su fórmula concisa, las reglas difusas *Si-entonces* se emplean siempre para capturar los modos imprecisos de razonamiento, los cuales juegan un rol importante en la habilidad humana para tomar decisiones en un entorno de incertidumbre y de imprecisión.

Básicamente se consideran dos tipos principales de reglas difusas: *Mamdani* [40] y *Takagi–Sugeno* (TS) [39], dependiendo de la forma de consecuentes.

El tipo *Mamdani* utilizan conjuntos difusos como los consecuentes de reglas, mientras que el tipo TS emplea funciones (normalmente constantes o lineales) de las variables de entrada como consecuentes. Un ejemplo del tipo *Mamdani* es :

Si *la presión* es *alta*, entonces *el volumen* es *pequeño.*

Aquí *la presión* y *el volumen* son variables lingüísticas, *alta* y *pequeño* son valores lingüísticos o etiquetas que son caracterizadas por funciones de membresía.

La forma de *Si-entonces* en las reglas difusas propuesta por *Takagi- Sugeno* tiene conjuntos difusos nada más en la parte del antecedente. Por ejemplo:

Si *la velocidad* es *alta*, entonces *la fuerza* $= k * (velocidad)^n$.

Aquí *alta* en la parte de antecedente es una etiqueta lingüística, caracterizada por una función de membresía. Sin embargo la parte consecuente se describe por una ecuación no difusa de la variable de entrada, *velocidad.* Esta forma se llama *n-ésimo* orden TS; si la parte del consecuente nada más es una constante se denomina de *orden cero* TS [49]. Por ejemplo:

Si la *velocidad* es *alta*, entonces *la fuerza* $= k$=0.7.

Ambas clases de reglas difusas de *Si-entonces* se usan desde hace ya tiempo en el modelado y control extensivamente. A través del uso de etiquetas lingüísticas y funciones de membresía, las reglas difusas *Si-entonces* pueden fácilmente capturar el espíritu del razonamiento del ser humano.

2.2. Estructura de un SLD

Básicamente, un *SLD* consiste en cuatro bloques.(ver figura 2.2)

• **Base de conocimientos.** Este bloque contiene dos partes:

Base de reglas. Integrada por un cierto número de *Si-entonces* reglas difusas.

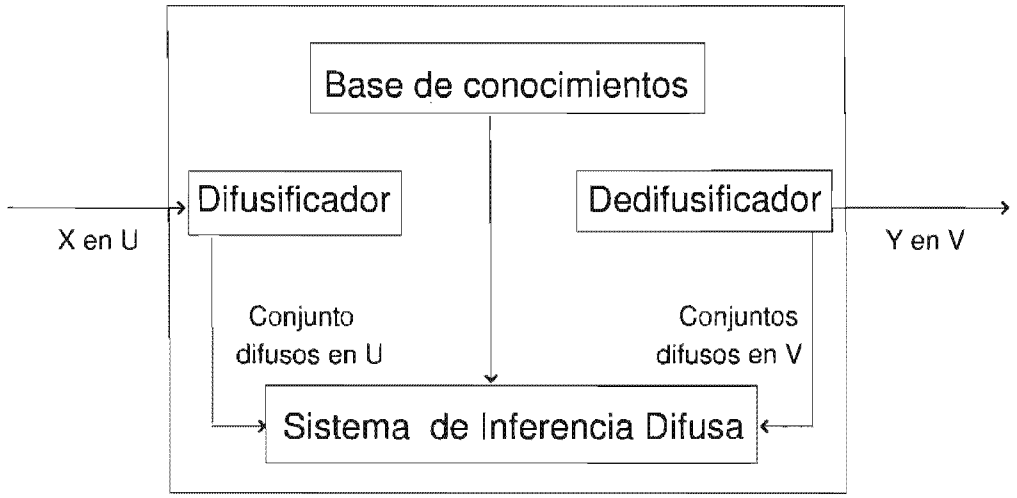

Figura 2.2: Estructura de un sistema de lógica difusa.

Base de datos. Contiene las funciones de membresía de conjuntos difusos usados en las reglas difusas.

- **Unidad de toma de decisiones.** Ejecuta las operaciones de inferencia en las reglas.
- **Interfase de difusificación.** Transforma entradas precisas a grados de correspondencia con las reglas difusas.
- **Interfase de dedifusificación.** Transforma los resultados difusos de la inferencia en salidas precisas.

2.2.1. La base de conocimientos

La base de conocimientos contiene dos partes, una es una base de reglas difusas la cual es una colección de reglas difusas *Si-entonces* :

$$R = [R^{(1)}, R^{(2)}, ..., R^{(M)}], \tag{2.10}$$

con

$$R^{(l)} : Si\ (x_1\ \ es\ F_1^l\ y\ ...y\ x_p\ es\ F_p^l);\ entonces\ (y_1\ es\ G_1^l, ..., y_q\ es\ G_q^l). \tag{2.11}$$

En la cual $\underline{x} = (x_1, ..., x_p)^T$ y $\underline{y} = (y_1, ..., y_q)^T$ son las entradas y salidas del sistema difuso. F_i^l y G_j^l son etiquetas de conjuntos difusos en U_i y V_j, respectivamente, $1 \leq p \leq m, 1 \leq q \leq n$, y $l = 1, 2, ..., M$. Entonces el sistema difuso ejecuta un mapeo de U_i a V_j. Esta es una descripción para un sistema difuso que tenga multi-entradas y multi-salidas. Obviamente, R^l de la regla anterior puede ser descompuesto en una colección de q reglas.

$$R^l = [R_1^l, R_2^l, ..., R_q^l] \tag{2.12}$$

Esta es una descripción para los sistemas difusos que tienen múltiple-entradas y una salida, donde

$$R_j^l : Si\ (\ x_1\ es\ F_1^l\ y\ \ldots\ x_p\ es\ F_p^l\ )\ ;\ entonces\ (\ y_j\ es\ G_j^l). \tag{2.13}$$

La parte de la premisa, cada regla difusa *Si-entonces* puede ser considerada como una descripción local del sistema.

La base de reglas difusas forma el núcleo del sistema de inferencia difusa. Cada regla difusa de la presentación (2.11) define una implicación difusa en la lógica multivaluada, aunque hay muchas interpretaciones de reglas difusa propuestas en la literatura de lógica difusa. Aquí se presentaran únicamente cuatro interpretaciones de reglas difusas que se usan frecuentemente [44]. Para simplificarlas, se denota $F_1^l \times \cdots \times F_n^l = A$ y $G^l = B$, entonces reglas $A \longrightarrow B$ pueden interpretar así:

1. Regla de Mínima implicación difusa:

$$\mu_{A \to B}(\underline{x}, y) = \text{mín}[\mu_A(\underline{x}), \mu_B(y)]. \tag{2.14}$$

2. Regla de Producto de implicación difusa:

$$\mu_{A \to B}(\underline{x}, y) = \mu_A(\underline{x})\mu_B(y). \tag{2.15}$$

3. Regla Aritmética de implicación difusa:

$$\mu_{A \to B}(\underline{x}, y) = \text{mín}[1, 1 - \mu_A(\underline{x}) + \mu_B(y)]. \tag{2.16}$$

4. Regla de Máxima implicación difusa:

$$\mu_{A \to B}(\underline{x}, y) = \text{máx}[\text{mín}[\mu_A(\underline{x}), \mu_B(y)], 1 - \mu_A(\underline{x})]. \tag{2.17}$$

donde el valor μ es llamado *fuerza* o *peso* de la regla; $\mu_A(\underline{x}) = \mu_{F_1^l \times \cdots \times F_n^l}(\underline{x})$ es definida por:

$$\mu_{F_1^l \times \cdots \times F_n}(\underline{x}) = \mu_{F_1^l}(x_1) \star \cdots \star \mu_{F_n^l}(x_n). \tag{2.18}$$

donde " $\star$ " denota la norma-T[1].

La otra parte de la base de conocimientos es una base de datos la cual define las funciones de membresía del conjunto difuso que se usarán en las reglas difusas.

Como se describe en la parte anterior, para definir las funciones de membresía, primero se considera el tipo de función de membresía; y en segundo término son los parámetros para las funciones de membresía seleccionadas. Por ejemplo: para la función de membresía triangular, se necesita definir su rango y su punto máximo. Desde esta función de membresía se puede truncar una función de membresía trapezoidal, la cual tiene una cima horizontal; estas funciones de membresía que se forman en líneas rectas tiene la ventaja de simplicidad. Las funciones de membresía gaussiana y campana (esta última función tiene un parámetro más que la función de membresía gaussiana), son métodos más populares para especificar conjuntos difusos, por su suavidad, notación concisa y rango infinito.

Aunque las funciones de membresía gaussiana y las de campana son suaves, ellas no pueden especificar funciones de membresía asimétricas, que son importantes en ciertas aplicaciones. Sin embargo se cuenta con las funciones de membresía sigmoides, que están abiertas a derecha o izquierda, se pueden sintetizar las funciones de membresía asimétricas y cerradas.

Para realizar diferentes tareas se pueden crear otras funciones de membresía definidas de forma distinta a las presentadas, aunque normalmente con los cuatro tipos de fun-

[1]Una norma-T, denotado por $*$, es una función R^2 desde [0, 1] $\times$ [0, 1] a [0, 1], la cual incluye la intersección difusa, el producto algebraico y el producto limitado,etc. definida como:

$$\begin{array}{ll} \min\{x, y\} & \text{intersección difusa} \\ xy & \text{producto algebraico} \\ x * y = \max\{0, x + y - 1\} & \text{producto limitado} \end{array} \tag{2.19}$$

ciones de membresía presentadas se pueden realizar con la mayoría de las tareas [37].

2.2.2. Inferencia difusa

La parte de "la unidad de toma de decisiones"se que presenta como el sistema de inferencia difusa en la figura 2.2 ejecuta las operaciones de inferencia basada en las reglas difusas *Si-entonces*. Estas reglas forman la base de conocimientos. El proceso de inferencia difusa incluye todas las piezas que hemos visto anteriormente: las funciones de membresía, operadores de lógica difusa y reglas *Si-entonces*.

Los pasos del razonamiento difuso son [37]:

1) Comparar las variables de entrada con las funciones de membresía en la parte de premisa para obtener los valores de pertenencia de cada entrada con respecto a cada conjunto difuso. Este paso se denomina **difusificación**.

2) Combinar (a través de un operador norma-T específica, normalmente multiplicación o min.) los valores de funciones de membresía en la parte de premisa para obtener el peso de cada regla.

3) Generar la parte consecuente (difuso o preciso) de cada regla dependiendo del peso.

4) Agregar el consecuente ponderado para producir una salida precisa. Este paso se denomina **dedifusificación**.

2.2.3. Difusificador

Un difusificador mapea un punto preciso $\underline{x} = (x_1..., x_n)^T \in U$ a un conjunto difuso A_x en U. Intuitivamente, es un mapeo de números a palabras descriptivas. Por ejemplo, se podría usar un conjunto difuso de pesos con valores lingüísticos (muy-ligero, ligero, medio, pesado, muy-pesado) que nos permita escribir reglas en etiquetas de palabras descriptivas, con el objetivo de que sea más fácil de entender, no en las etiquetas de valores numéricos.

Ahora necesitamos una manera de transformar los valores de su grado de pertenen-

cia en un conjunto difuso de palabras descriptivas. Este proceso se llama difusificación y se hace por medio de una función de membresía: se comparan las variables con las funciones de membresía en la parte de premisa y se obtienen los valores de membresía como se puede ver en la figura 2.3.

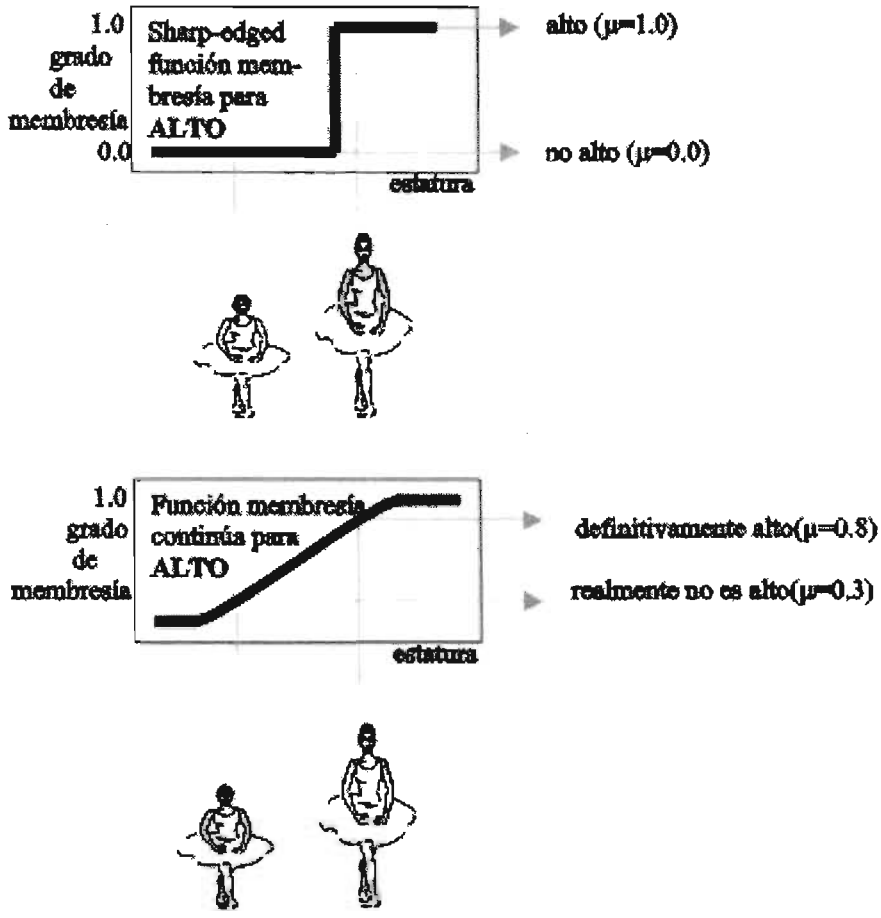

Figura 2.3: Un ejemplo de funciones de membresía que ilustra cómo obtener el valor.

Hay dos tipos de difusificador [45]:

1) Difusificador de singleton:

A_x es un singleton difuso con el soporte $\underline{X}$, es decir, $\mu_{A_x}(\underline{X'}) = 1$ para $\underline{X'} = \underline{X}$ y $\mu_{A_x}(\underline{X'}) = 0$ para cualquier otro $\underline{x'} \in U$ con $\underline{X'} \neq \underline{X}$.

2) Difusificador de nonsingleton:

$\mu_{A_x}(\underline{X}) = 1$ y $\mu_{A_x}(\underline{X'})$ disminuye desde 1 cuando X' se mueve desde $\underline{X}$. Por ejemplo,

$\mu_{A_x}(\underline{X'}) = \exp[-\frac{(\underline{X'}-X)^T(X'-X)}{\sigma^2}]$, donde σ^2 es un parámetro que caracteriza la forma de $\mu_{A_x}(\underline{X'})$.

En la literatura, se usa más el difusificador de singleton, pero cuando las entradas están corrompidas por ruidos, se considera que el difusificador de nonsingleton sería útil [43].

2.2.4. Dedifusificador

Un dedifusificador consiste en convertir un valor lingüístico en numérico. Es decir mapea un conjunto difuso en V a un punto preciso en V. Este proceso se llama dedifusificación y es el proceso opuesto al de difusificación. Se tiene un conjunto difuso de palabras descriptivas, y se espera convertir estos a valores reales para que los usuarios puedan recibir una salida de números. En este caso, se podría decir que la dedifusificación y la difusificación son reversibles. El dedifusificador es necesario porque para muchas aplicaciones, a un sistema difuso es necesario darle una salida precisa, no importa si se usa como un controlador, un identificador, etc., porque la salida de inferencia difusa es un conjunto difuso $A \circ R$ en V, el dedifusificador cambia $A \circ R$ a un punto preciso $y \in V$.

Normalmente tenemos al menos tres selecciones para este mapeo.

1)Dedifusificador máximo:

$$y = \underset{\sup_{y^l \in R}}{\arg} (\mu_{A_x \circ (R^{(1)}, \cdots, R^{(M)})}(y^l)) \tag{2.20}$$

donde $A_x \circ (R^{(1)}, \cdots, R^{(M)})$ es el conjunto final definido por todas las M reglas en la base de reglas difusas, el cual se obtiene al combinar $\mu_{A_x \circ (R^{(1)}, \cdots, R^{(M)})}(y^l)$ usando la disyunción difusa:

$$\mu_{A_x \circ (R^{(1)}, \cdots, R^{(M)})}(y^l) = \mu_{A_x \circ R^{(1)}}(y) \dot{+} \cdots \dot{+} \mu_{A_x \circ R^{(M)}}(y) \tag{2.21}$$

2) Dedifusificador centroide: éste es el método más común en la literatura, dado por

$$y = \frac{\sum_{l=1}^{M} \bar{y}^l (\mu_{A_x \circ R^{(l)}}(\bar{y}^l))}{\sum_{l=1}^{M} (\mu_{A_x \circ R^{(l)}}(\bar{y}^l))} \tag{2.22}$$

donde $\bar{y}^l$ es un punto en R en el cual $\mu_{G^l}(y)$ alcanza su máximo valor, y $\mu_{G^l}(y) = \mu_{A_x \circ (R^{(1)},\cdots,R^{(M)})}(y^l)$.

3) Dedifusificador centroide modificado:

$$y = \frac{\sum_{l=1}^{M} \bar{y}^l (\mu_{A_x \circ R^{(l)}}(\bar{y}^l)/\sigma^l)}{\sum_{l=1}^{M} (\mu_{A_x \circ R^{(l)}}(\bar{y}^l)/\sigma^l)} \tag{2.23}$$

donde σ^l es un parámetro para caracterizar la forma de $\mu_{G^l}(y)$; para que la forma de $\mu_{G^l}(y)$ sea estrecha, σ^l debe ser pequeña. Por ejemplo, si $\mu_{G^l}(y) = \exp[-(\frac{y-y^l}{\sigma_l})^2]$, entonces σ_j^l es el parámetro mencionado.

2.3. Diseño de SLDs

La lógica difusa es un paradigma para una metodología de diseño alternativo, la cual puede ser aplicada para desarrollar tanto sistemas lineales como no lineales. Debido al uso de la lógica difusa, los diseñadores pueden realizar desarrollos a menor costo con características superiores y mejor desempeño de los productos finales. Unos subclases de sistemas difusos se ilustran en la figura 2.4.

2.3.1. Metodología de diseño

Los sistemas difusos se pueden diseñar por entrevista a un experto con el fin de transformar su conocimiento implícito de procesos fundamentales a un conjunto de variables lingüísticas y reglas difusas. Particularmente para una tarea compleja de control, obtener la base de conocimiento difuso a partir de un experto a veces es tedioso y no es confiable, ya que se aplica la metodología de prueba y error.

Clásicamente se consideran estos pasos para diseñar un *SLD*:

Paso 1: Definición de las señales de **entrada/salida**. Es decir, cuáles variables del sistema dinámico se tomarán como señales de entrada y salida del sistema. Las variables aquí se presentan como variables lingüísticas, así como también las variables de salidas. Además la selección de variables lingüísticas propias para formular reglas difusas es un factor muy importante en la ejecución y el desempeño del sistema. El conocimiento

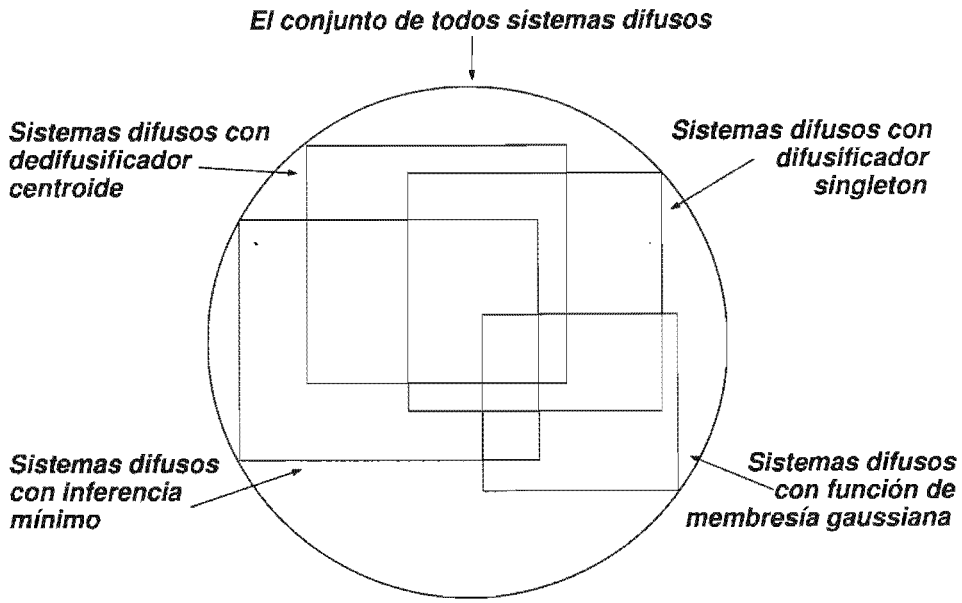

Figura 2.4: Subclases de sistemas difusos.

empírico y la intuición de ingeniería juegan un papel importante al seleccionar las variables lingüísticas y sus funciones de membresía correspondientes. Estas variables transforman los valores numéricos de las entradas del sistema a su correspondiente valor difuso. El número de estas variables lingüísticas especifica el tipo del problema que puede lograrse usando el sistema difuso. Si el número de variables lingüísticas incrementa, el tiempo computacional y la memoria necesaria se incrementan de forma exponencial. Entonces debido al compromiso entre la complejidad del problema y el tiempo computacional es necesario escoger un número adecuado de variables lingüísticas.

Paso 2: Definir los **conjuntos difusos** para cada variable lingüística, y sus **funciones de membresía** correspondientes.

Básicamente la sensibilidad de una variable del problema determina el número de subconjuntos difusos. Por ejemplo si tenemos una variable lingüística *estatura* en el problema que se mencionó en el principio del capítulo "¿Es Juan alto?", dos conjuntos

difusos *alto, bajo* serán suficientes. La misma variable *estatura* en el problema de investigar cómo los alimentos nutritivos afectan la estatura, debería tomar más conjuntos difusos. Para esta variable podría ser *muy alto, alto, medio, bajo, muy bajo.*

Después de especificar los conjuntos difusos para cada variable lingüística de las entradas y salidas, se necesitan definir las funciones de membresía. Se consideran dos pasos: primero se necesita seleccionar el tipo de funciones de membresía. Por ejemplo: triangular, gaussiana, sigmoide etc. y segundo definir los parámetros que corresponden a dichas funciones de membresía.

Determinar las funciones de membresía

Como ya se describió antes, un conjunto difuso es totalmente caracterizado por sus funciones de membresía. Para la mayoría de las aplicaciones, los conjuntos definidos tendrán que ser fáciles e identificables. Para otras aplicaciones tendrán que ser determinados por adquisición de conocimiento de un experto o un grupo de expertos. Una vez que los nombres de estos conjuntos difusos han sido establecidos, se tienen que considerar sus funciones de membresía asociadas.

El método adoptado para adquirir la forma de las funciones de membresía siempre depende de la aplicación. Para muchos problemas de control, la suposición es que las funciones de membresía son de forma triangular o gaussiana. Los parámetros normalmente son obtenidos con base en la experiencia del ingeniero de control y son generados automáticamente. Pero para otras aplicaciones, estas funciones de membresía quizás no son adecuadas, entonces tendrán que ser obtenidas por expertos, por un método estadístico o por generación automática de las formas. La determinación de una función de membresía puede ser entonces calificada de forma manual o automática. El intento y el interés de este campo lleva a la consideración de que las técnicas de Inteligencia Artificial pueden ser usadas para asistir al desarrollo de sistemas de inferencia difusos.

Método manual para determinar las funciones de membresía: En la literatura están reportadas varias técnicas estadísticas para determinar funciones de membresía. Estas técnicas caen en dos categorías generales [47]: *Uso con frecuencia* o por *estimación*

directa. El método de frecuencia obtiene una función de membresía para medir el porcentaje de gente en un grupo (típicamente expertos en un campo particular) que responde que sí a una pregunta, sobre si un objeto pertenece a un conjunto particular. Los métodos de estimación directa toman una manera diferente para preguntar a los expertos cómo graduar un evento en una escena.

Métodos automáticos para determinar las funciones de membresía: La generación automática de funciones de membresía cubre una amplia variedad de métodos diferentes. Esencialmente, la diferencia entre la generación automática y los métodos manuales es que en el primero el experto es completamente eliminando por el proceso y en el segundo las funciones de membresía se ajustan basándose en una conjetura inicial por el experto. Para que no se confundan con aplicaciones o métodos para clasificar, aquí se describen dos técnicas usadas - redes neuronales y algoritmos genéticos (el énfasis está en el uso de técnicas de la computación suave moderna).

Redes neuronales artificiales:

Redes neuronales artificiales (*ANNs* por su sigla en Inglés) [11] son técnicas basadas en la manera en que el cerebro procesa información. Esta técnica permite la estimación de funciones no-lineales particularmente. Hay mucha literatura en la teoría y aplicación de *ANNs*. Aquí nos interesa el uso de redes neuronales para ayudar al ingeniero en el desarrollo de los *SLDs*.

Takagi y *Hayashi* [38] señalaron que el razonamiento difuso presenta los siguientes problemas particulares:

1. La falta de un método definido para determinar las funciones de membresía.
2. La falta de una función de aprendizaje.

Para resolver estos problemas propusieron usar *ANNs*. El método consiste en investigar reglas *Si-entonces* usando *ANNs* para determinar las funciones de membresía del antecedente, para después determinar los componentes del consecuente como la salida para cada regla. Para emplear dicho método hay que obtener pares de datos (en el caso de control), aplicar un algoritmo de clasificación (clustering) para agrupar los datos y

aplicar una *ANN* a estos datos agrupados para determinar una función de membresía de un patrón en conjuntos difusos particulares.

Esta propuesta fue utilizada en dos problemas: En la estimación de la densidad demandada de oxígeno químico [2] en Ōsaka Bay y en la estimación de la superficie de una cerámica [49]. El método en ambos casos resultó ser muy adecuado, porque esta combinación de redes neuronales y razonamiento difuso no solamente permite la generación automática de funciones de membresía en ciertas aplicaciones sino también del uso de reglas difusas *Si-entonces*.

Como se describió antes, usar conocimiento de expertos es la manera más común para determinar funciones de membresía. Otros métodos [46] construyen con la pericia proporcionada por un experto y luego usan *ANNs* para ajustar adecuadamente la función de membresía. En otras palabras, la pareja (X, y) que describe la relación entre X y y son presentadas a la red neuronal la cual mete aproxima una función a los puntos. Para proveer mejor interpolación entre los puntos se usa el algoritmo de retropropagación.

Entre la lógica difusa y las redes neuronales existe una sinergia atractiva (esto se verá en el capítulo 3), y el uso de *ANNs* para determinar funciones de membresía es solamente un área en donde esta técnica puede ayudar.

Algoritmos Genéticos

Los algoritmos genéticos(AGs) proporcionan una técnica de búsqueda basada en la genética natural. Este método adopta algunas ideas de la evolución natural para representar datos como cromosomas y genes, y ejecutar operaciones tales como cruza y mutación.

Meredith y sus colaboradores [27] han aplicado algoritmos genéticos para ajustar funciones de membresía en un controlador difuso para un helicóptero. Inicialmente, un ingeniero de control manejaba las funciones de membresía y luego usa un algoritmo genético para ajustar los parámetros y definirlos, usándolos para minimizar el

[2] chemical oxygen demand density

movimiento del helicóptero en vuelo.

Los AGs ofrecen una alternativa no convencional para obtener funciones de membresía. Los dos métodos se explicarán posteriormente a detalle.

Paso 3: Las reglas difusas *Si–entonces.*

Las reglas difusas definen la relación entre las entradas y salidas del sistema. Como las reglas son definidas por variables lingüísticas, entonces para definir el antecedente, se considera necesario seleccionar entre los operadores lógicos *min, max* y *not*, (*min* se usa frecuentemente) según el problema que necesitamos resolver. Luego se considera cuántas reglas son adecuadas. Una regla obviamente no es suficiente, pero tampoco se puede decir si será mejor un número de reglas es grande. La complejidad del problema depende del número de conjuntos difusos que corresponden a cada variable.

Ahora se necesita buscar la región difusa de las salidas para cada regla. Hay métodos diferentes para buscar las salidas. Los métodos Mínimo–máximo y Máximo-min como se pueden ver en las figuras 2.5 y 2.6, son unos de los más importantes.

Si las reglas difusas son conectadas usando el operador Y definido previamente, entonces el operador Y busca el valor mínimo entre dos funciones de membresía, por lo que el operador Y se usa para obtener el valor mínimo entre las funciones de membresía de entrada. Luego se busca el mínimo entre este resultado y las funciones de membresía de salidas. Finalmente las funciones de membresía de salida de cada regla se calculan, como se puede ver en la figura 2.5. En el caso de considerar el operador *max* para calcular el valor de la función de membresía de salida, entonces obtiene el valor máximo. Ver la figura 2.6.

Paso 4: Dedifusificación.

Debido a que se necesita una señal no-difusa para un sistema, se tendrá que determinar un valor numérico. Se consideran muchas técnicas diferentes para la dedifusificación de las cualidades difusas, tal como se describió antes entre las cuales están el método del máximo, y el método del centroide.

En la figura 2.6 se representa el método centroide. En este método de dedifusifi-

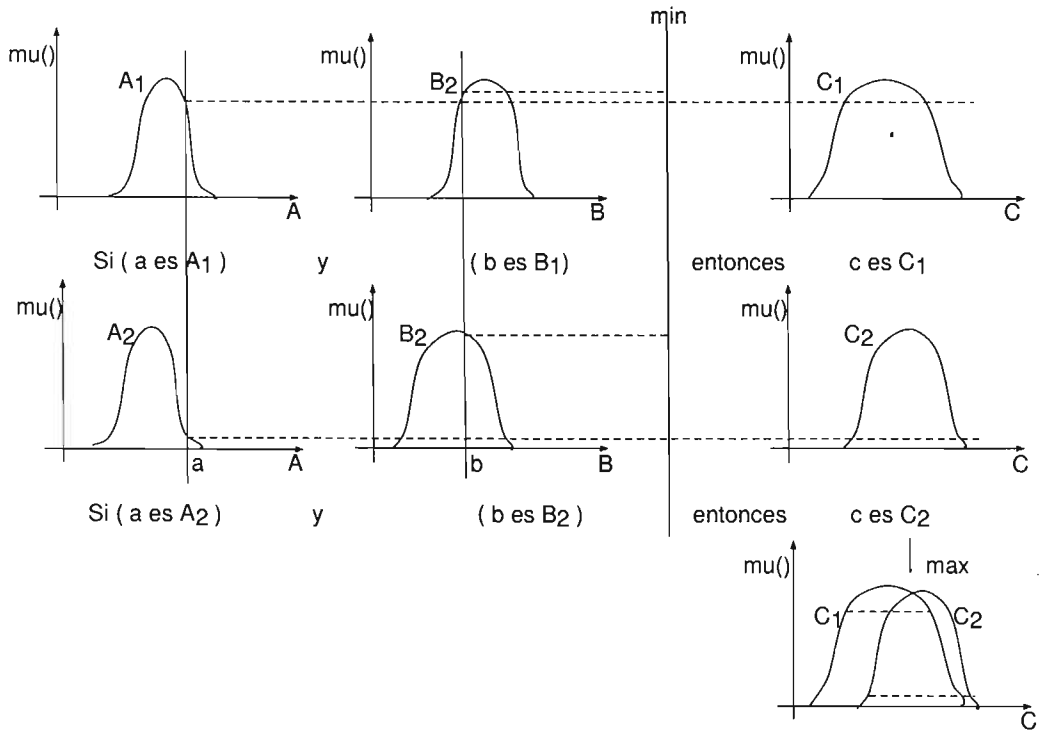

Figura 2.5: La descripción del método Min-Max

cación, el promedio pesado de las funciones de membresía o el medio de gravedad de área limitada por la curva de la función de membresía son calculados por el valor preciso más típico de la calidad difusa.

$$\bar{y} = \frac{\int y\mu(y)dy}{\int \mu(y)dy} \tag{2.24}$$

2.3.2. Una comparación entre diseño difuso y diseño convencional

Para reconocer porqué una metodología de diseño basado en el paradigma difuso es muy atractivo en las aplicaciones, se presenta un ejemplo de una aplicación de control implantado en [50]. Ahora vamos a examinar un típico diseño de flujo. En la figura 2.7 se ilustra la secuencia de los pasos necesarios de un diseño para desarrollar un

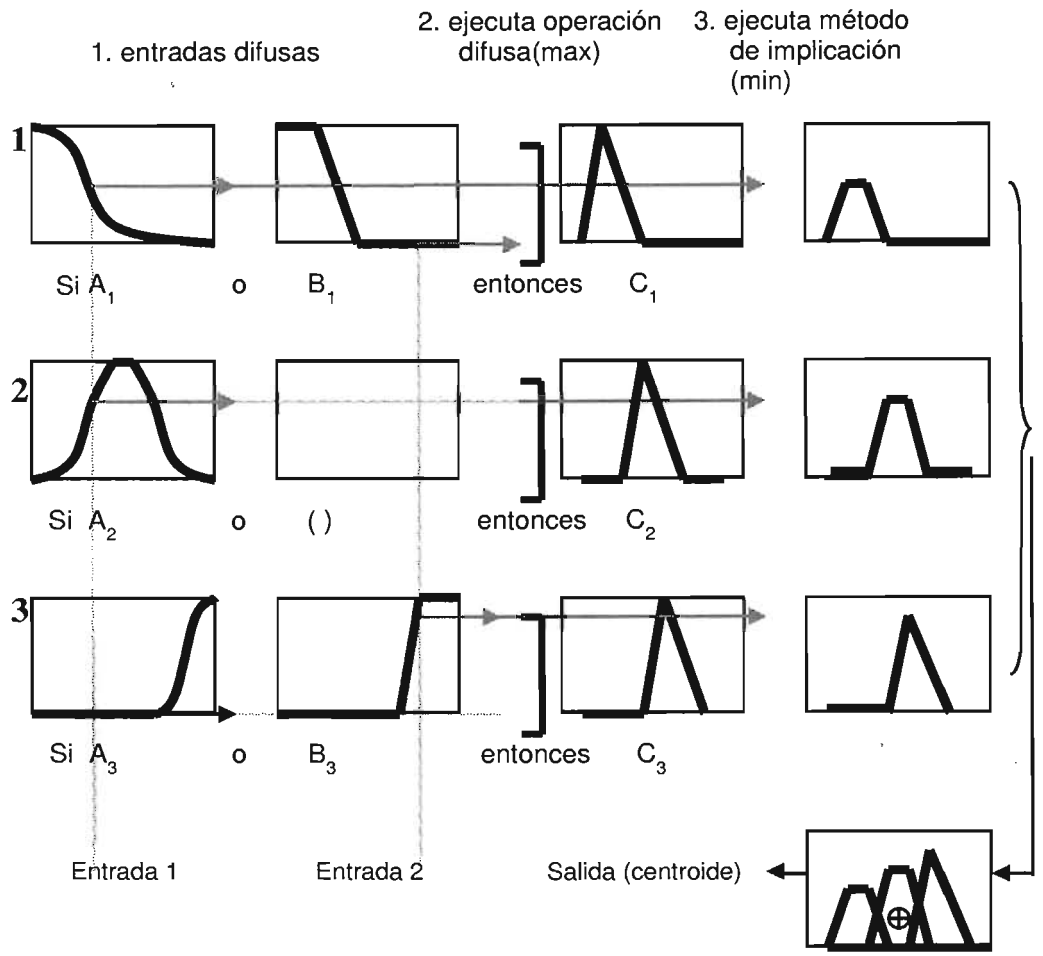

Figura 2.6: La descripción del método Max-Min y el método centroide

controlador usando un método convencional y un método difuso.

Usando el método convencional, nuestro primer paso es entender el sistema físico y sus requisitos de control. Basado en esta comprensión, el segundo paso es desarrollar un modelo que incluya la planta, sensores y actuadores. El tercer paso es usar teoría lineal de control para determinar una versión simplificada del controlador, tal como el controlador PID[3]. El cuarto paso es desarrollar un algoritmo para el controlador simplificado. El último paso es simular el diseño incluyendo los efectos de no-linealidad,

[3]Proporcional, Integral y Derivativo

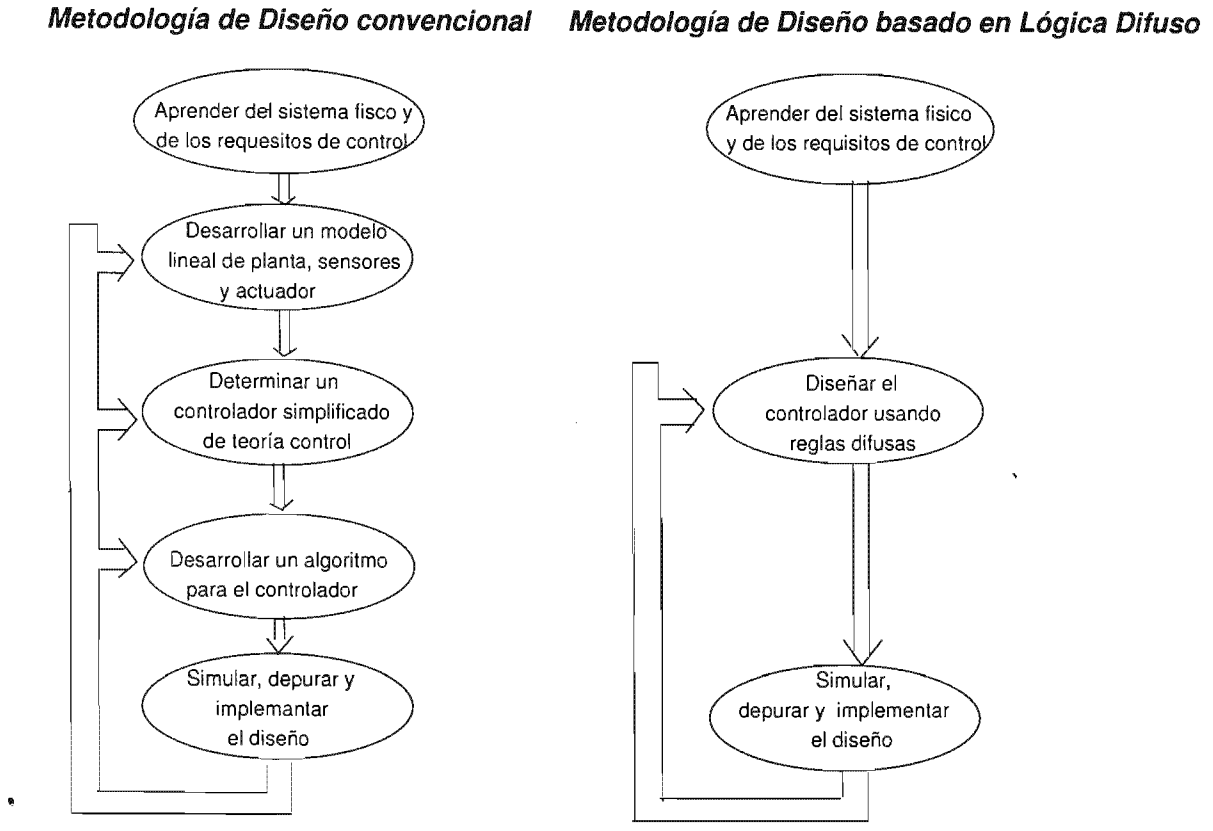

Figura 2.7: Diseño convencional y diseño difuso.

ruido y parámetro de variación. Si el desempeño no es satisfactorio, necesitamos modificar el modelo de nuestro sistema, rediseñar el controlador, reescribir el algoritmo y reintentarlo.

Por otro lado, con lógica difusa el primer paso es comprender y caracterizar el comportamiento del sistema usando nuestro conocimiento y experiencia. El segundo paso es diseñar el algoritmo de control directamente usando reglas difusas, las cuales describen los principios de las regulaciones del controlador en cuanto a las relaciones entre sus entradas y sus salidas. El último paso es simular y ajustar el diseño; si el desempeño no es satisfactorio, nada más necesitamos modificar algunas reglas difusas y reintentarlo.

Aunque las dos metodologías de diseño son similares, la metodología basada en el

paradigma difuso simplifica el diseño sustancialmente. Esto resulta en muchos beneficios significativos, tales como: reducir el tiempo de desarrollo, simplificar el diseño y el tiempo más corto para entrar al mercado.

Capítulo 3

Sistemas Adaptables

3.1. Definición y mecanismos de la adaptación

Un sistema adaptable es un sistema que cambia su comportamiento para responder al cambio de su ambiente. Un cambio que ocurre siempre es pertinente para lograr un objetivo. Podemos decir que las plantas, los animales, el ser humano, la sociedad, hasta el universo, son adaptables.

Un ejemplar de adaptación es el crecimiento de una planta con unos obstáculos en su alrededor. Sin obstáculos una planta crecerá siguiendo un tipo de patrón. Pero el crecimiento cambiará si hay obstáculos, y el cambio específico que ocurre depende de la disposición de obstáculos. Digamos que esta planta se adaptó a su ambiente por la evolución.

Otro tipo de comportamiento adaptable, el aprendizaje, es típicamente más asociado con animales. Un animal aprende desde sus experiencias porque cambios en el cerebro (impresión de memorias) cambian el comportamiento del animal en el futuro.

Generalmente, el aprendizaje ocurre cuando un patrón de comportamiento del sistema cambia como un resultado de una interacción con el ambiente. Esto incluye dos niveles de respuesta por el sistema: el sistema responde yendo por una trayectoria y la trayectoria se ve alterado por el cambio del sistema.

Generalmente asociamos la adaptación con comportamiento de búsqueda de meta. Un sistema de búsqueda de meta intenta adaptarse (cambiar su comportamiento) para lograr su meta.

Un controlador automático de nivel de líquido acoplado a un sistema de tanques tiene comportamiento de búsqueda de meta. Puede decirse que es adaptable, porque cambios en el ambiente o en el sistema en sí mismo cambian su comportamiento. El cambio es una diferencia en tiempo real de la salida del controlador la cual lleva a cabo la meta del sistema, que es un nivel deseado.

En sistemas físicos, como el fluido siguiendo la gravedad, también se presenta el comportamiento de búsqueda de meta [51]. El agua fluye por todos lados y pasa obstáculos. El comportamiento de búsqueda de meta incluye retroalimentación por las interacciones colectivas de las moléculas del agua y la tensión de superficie.

Los sistemas químicos de equilibrio integrados por sustancias reactivas [51] también se adaptan a cambios en una manera bien comprensiva. Si se añade uno de los reactantes, algunas sustancias químicas reaccionarán para reducir el cambio en concentración de las sustancias químicas añadidas: el proceso químico reducirá al que se le impone.

Una colección de organismos puede responder también al cambio ambiental. La evolución es la respuesta de un grupo de organismos que ocurre sobre muchas generaciones. Los cambios evolutivos reflejan la respuesta de la colección de organismos a su ambiente. Este incluye ambos, cambios internos (ejem. en el genotipo) y cambios externos (ejem. en el fenotipo).

Se usan diferentes herramientas para realizar la tarea de adaptación, tales como *ecuaciones diferenciales y en diferencias*, la *teoría de autómatas* y la *teoría de sistemas no lineales*. En la literatura recién publicada, la *teoría de estimación por redes neuronales*, la *teoría de lógica difusa* y la *teoría de optimización por* AGs son ampliamente estudiadas.

3.2. Sistemas adaptables basados en la lógica difusa

Las razones por las que construyen sistemas adaptables basados en la lógica difusa son de dos categorías: teóricas y prácticas.

Las razones teóricas son:

- Como una regla general, un buen método de ingeniería debería poder hacer uso efectivo de toda la información obtenible. Si el modelo matemático de un sistema es tan difícil de obtener (esto es cierto para muchos sistemas), entonces, como lo mencionamos aquí, la información más importante viene de dos orígenes: (1) sensores que proporcionan mediciones numéricas de variables principales. y (2) expertos humanos quienes proporcionan descripciones lingüísticas sobre instrucciones del sistema. Por su diseño, un sistema difuso provee un marco sistemático y eficiente para incorporar información lingüística de expertos humanos. Sin embargo, en sistemas convencionales, no se puede incorporar información lingüística a sus diseños. Si en alguna ocasión la información más importante viene de expertos humanos, entonces un sistema difuso es la mejor selección.

- Un sistema difuso es un método libre de modelo; es decir, no se necesita un modelo matemático del sistema bajo su aplicación. Ahora, los ingenieros enfrentan sistemas bajo aplicaciones cada vez más complejas, y los modelos matemáticos de estos sistemas son cada vez más difícil de obtener. Por lo tanto los métodos que no usan modelos serán cada vez más importantes en la ingeniería. En el caso de control, el control convencional también tiene unos métodos libres de modelos, tal como control PID. Pero un sistema difuso proporciona otra alternativa.

- Un sistema difuso aproxima un sistema no lineal, el cual es bien justificado debido al teorema de Aproximación Universal [43]. Este teorema explica que estos sistemas de lógica difusa son suficientes para ejecutar generalmente cualquier acción no lineal. Por lo tanto, si se seleccionan los parámetros de sistemas difusos en forma adecuada, siempre es posible diseñar un sistema difuso conveniente para

el sistema no lineal bajo las aplicaciones dadas.

En las razones teóricas anteriores, un sistema difuso es considerado como una teoría y es evaluado desde un punto de vista teórico. Pero se sabe que la teoría y la práctica tienen diferente énfasis. Por ejemplo, la teoría enfatiza generalidad y rigor, sin embargo la práctica enfatiza aplicabilidad a problemas particulares.

Las razones prácticas para usar un sistema difuso son:

- Un sistema difuso es fácil de entender. Estos porque el sistema difuso emula la estrategia del humano, su principio es fácil de entender para personas que no son especialistas en ingeniería. Durante las últimas tres décadas, la teoría de sistemas convencionales ha usando herramientas matemáticas cada vez más avanzadas. Esto es necesario para resolver problemas difíciles en una manera rigurosa; sin embargo, esto también resulta en que cada vez menos ingenieros de la práctica puedan entender la teoría. Por lo tanto, los ingenieros de práctica que están en la primera línea de diseño de productos de consumo prefieren usar los métodos que son simples y fáciles de entender. Un sistema difuso es justamente un método de este tipo.

- Un sistema difuso es simple de implantar. Admite un alto grado de implementación paralela. Muchos fragmentos de VLSI difuso han sido desarrollados [42] [41], los cuales hacen la implementación de un sistema difuso más fácil y simple.

- Desarrollar un sistema difuso es barato. Desde un punto de vista práctico, el costo de desarrollo es uno de los más importantes criterios para un producto exitoso. Un sistema difuso es fácil de entender para ello, el tiempo para estudiar este método es corto; es decir el *costo de software* es bajo. Además como un sistema difuso es simple de implementar, el *costo de hardware* también es bajo. Adicionalmente, hay herramientas de software disponibles para diseñar sistemas difusos. Por lo tanto un sistema difuso es un método que tiene una alta proporción de beneficio/costo.

Los sistemas difusos se suponen trabajan en situaciones donde hay una gran incertidumbre o variación desconocida en parámetros de plantas y estructuras. Generalmente, el objetivo básico de un sistema adaptable es mantener la ejecución consistente en la presencia de estas incertidumbres. Por lo tanto, un sistema difuso avanzado debería ser adaptable.

Si un sistema de lógica difusa es equipado con un algoritmo de adaptación, se denomina un sistema difuso adaptable. Un sistema difuso adaptable puede ser construido con un único sistema difuso adaptable, o con varios sistemas difusos adaptables.

3.3. Algoritmo de adaptación

Debido a que los *SLDs* ya lo presentamos en el capítulo 2, en esta sección se enfoca a los algoritmos que se usan frecuentemente en el entrenamiento de un *SLD*.

3.3.1. Algoritmo de retropropagación

Este algoritmo es un poderoso algoritmo de entrenamiento, se puede aplicar en cualquier red de tipo *feedforward*. Por lo cual si podemos representar *SLDs* como una red de tal tipo, podemos usar la idea de retropropagación (back-propagation) para entrenarlos.

Consideramos que los *SLDs* pueden ser representados como una red de alimentación hacia adelante con tres capas como la presentada en la figura 3.1. Con la representación de *SLDs*, como una red, de manera directa se puede aplicar la idea de retropropagación para ajustar los parámetros $\bar{y}^l$, $\bar{x}_i^l$ y σ_i^l que están en la ecuación 3.1 la cual representa un sistema difuso.

Supongamos que tenemos un par entrada-salida $(\underline{x}^p, d^p)$, $\underline{x}^p \in U \subset R^n$, $d^p \in V \subset R$; nuestra tarea es determinar un *SLD* $f(\underline{x})$ de la forma

$$f(\underline{x}) = \frac{\sum_{l=1}^{M} \bar{y}^l [\prod_{i=1}^{n} a_i^l \exp(-(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)]}{\sum_{l=1}^{M} [\prod_{i=1}^{n} a_i^l \exp(-(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)]} \tag{3.1}$$

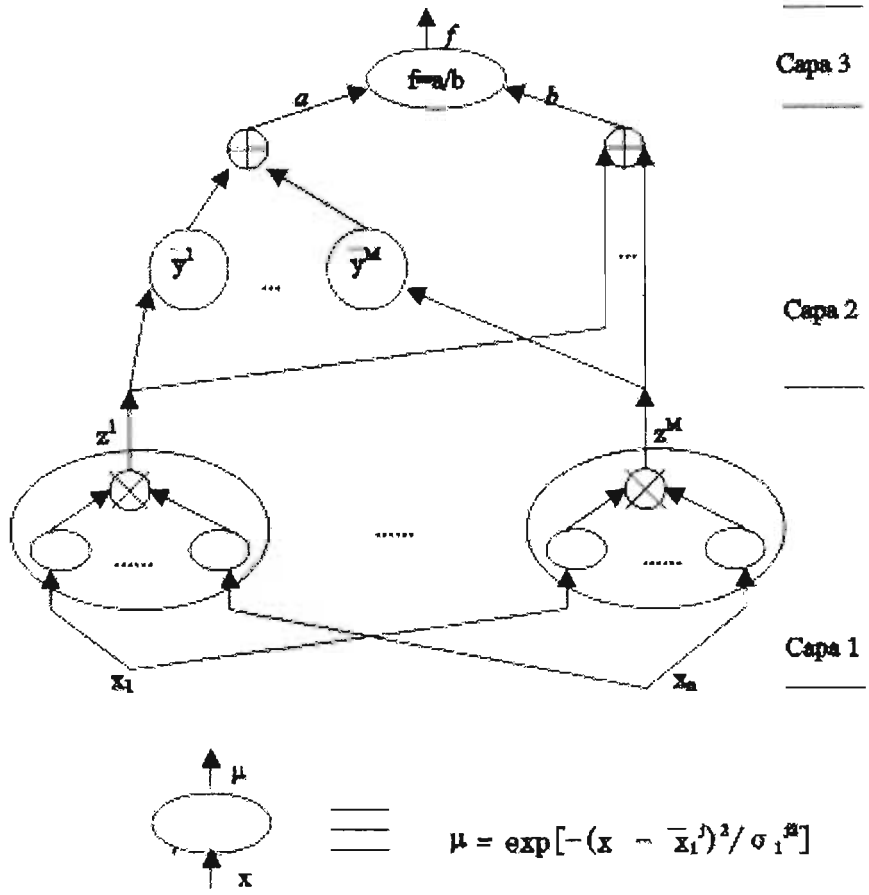

Figura 3.1: Representación en red de *SLD*[43]

la cual representa un sistema difuso con dedifusificador centroide, reglas de inferencia producto, difusificador de singleton y función de membresía gaussiana.

Para minimizar el error dado por la expresión:

$$e^p = \frac{1}{2}[f(\underline{x}^p) - d^p]^2 \tag{3.2}$$

Supongamos que los pesos $a_i^l = 1$ y M está dado, entonces el problema se convierte en entrenar los parámetros $\bar{y}^l$, $\bar{x}_i^l$ y σ_i^l tal que minimizen al error e^p. Posteriormente se usan e, f, y d para denotar e^p, $f(\underline{x}^p)$ y d^p, respectivamente.

Para entrenar $\bar{y}^l$ usamos el algoritmo de gradiente definido por la ecuación en difer-

encia:

$$\bar{y}^l(k+1) = \bar{y}^l(k) - \alpha \frac{\partial e}{\partial \bar{y}^l}\bigg|_k \tag{3.3}$$

donde $l = 1, 2, \cdots, M$, $k = 0, 1, 2, \cdots$, y α es un constante de medida de paso. Por la figura 3.1, podemos ver que f, por lo tanto e, dependen en $\bar{y}^l$ nada más de a, donde $f = a/b$, $a = \sum_{l=1}^{M}(\bar{y}^l z^l)$, $b = \sum_{l=1}^{M} z^l$, y $z^l = \prod_{i=1}^{n} \exp(-(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)$. Por lo tanto, usando la regla de la cadena tendremos

$$\frac{\partial e}{\partial \bar{y}^l} = (f-d)\frac{\partial f}{\partial a}\frac{\partial a}{\partial \bar{y}^l} = (f-d)\frac{1}{b}z^l \tag{3.4}$$

sustituyendo (3.4) en (3.3), obtenemos el algoritmo de entrenamiento para $\bar{y}^l$:

$$\bar{y}^l(k+1) = \bar{y}^l(k) - \alpha \frac{f-d}{b} z^l \tag{3.5}$$

donde $l = 1, 2, \cdots, M$, $k = 0, 1, 2, \cdots$.

Para entrenar $\bar{x}_i^l$, usamos

$$\bar{x}_i^l(k+1) = \bar{x}_i^l(k) - \alpha \frac{\partial e}{\partial \bar{x}_i^l}\bigg|_k \tag{3.6}$$

donde $i=1, 2, \cdots, n$, $l = 1, 2, \cdots, M$, y $k = 0, 1, 2, \cdots$. En la figura 3.1 podemos ver que f, y por lo tanto e, dependen en $\bar{x}_i^l$ nada más de z^l; por lo tanto, usando la regla de la cadena, tendremos

$$\frac{\partial e}{\partial \bar{x}_i^l} = (f-d)\frac{\partial f}{\partial z^l}\frac{\partial z^l}{\partial \bar{x}_i^l} = (f-d)\frac{\bar{y}^l - f}{b} z^l \frac{2(x_i^p - \bar{x}_i^l)}{\sigma_i^{l2}} \tag{3.7}$$

Sustituyendo la ecuación (3.7) en (3.6), obtenemos el algoritmo de entrenamiento para $\bar{x}_i^l$:

$$\bar{x}_i^l(k+1) = \bar{x}_i^l(k) - \alpha \frac{f-d}{b}(\bar{y}^l - f) z^l \frac{2(x_i^p - \bar{x}_i^l(k))}{\sigma_i^{l2}(k)} \tag{3.8}$$

donde $l = 1, 2, \cdots, M$, y $k = 0, 1, 2, \cdots$.

Usando el mismo método, obtenemos el siguiente algoritmo de entrenamiento para σ_i^l:

$$\sigma_i^l(k+1) = \sigma_i^l(k) - \alpha \frac{\partial e}{\partial \sigma_i^l}\bigg|_k = \sigma_i^l(k) - \alpha \frac{f-d}{b}(\bar{y}^l - f) z^l \frac{2(x_i^p - \bar{x}_i^l(k))^2}{\sigma_i^{l3}(k)} \tag{3.9}$$

donde i= *1, 2,* $\cdots$, n, $l = 1, 2, \cdots$, M, y $k = 0, 1, 2, \cdots$.

Los algoritmos de entrenamiento ejecutan un proceso de retropropagación del error. Al entrenar $\bar{y}^l$, el error normalizado $(f-d)/b$ es retropropagado a la capa de $\bar{y}^l$. Luego $\bar{y}^l$ se actualiza usando el algoritmo de entrenamiento de $\bar{y}^l$ en el cual z^l es la entrada a $\bar{y}^l$ (ver la figura 3.1). Para entrenar $\bar{x}_i^l$ y σ_i^l, el error normalizado $(f-d)/b$ multiplica $(\bar{y}^l - f)$ y z^l es retropropagado a la unidad de proceso de la capa 1, su salida es z^l. Entonces $\bar{x}_i^l$ y σ_i^l son actualizados usando su algoritmo de entrenamiento respectivamente, en los cuales las variables remantenidas $\bar{x}_i^l$, x_i^p y σ_i^l (que son las variables en el lado derecho de las ecuaciones 3.8, 3.9), pueden ser obtenidas localmente, excepto el error retropropagado $\frac{(f-d)}{b}(\bar{y}^l - f)z^l$.

El proceso de entrenamiento para el sistema difuso de la figura 3.1 es un proceso de dos pasos. Primero, para una entrada dada $\underline{x}^p$, hace cálculos hacia adelante a lo largo de la red *(SLD)* para obtener z^l $(l = 1, 2, \cdots,)$, a, b, y f. Luego entrena los parámetros de la red $\bar{x}_i^l$, x_i^p y σ_i^l $(i = 1, 2, \cdots, n, l = 0, 1, 2, \cdots M)$ para atrás usando las ecuaciones (3.5), (3.8), (3.9), respectivamente. Para las parejas múltiples de entrada-salida, dadas por $(\underline{x}^p, d^p)$ con $p = 1, 2, \cdots$, podemos entrenar el sistema para uno o más círculos de forward-backward para una pareja de entrada-salida antes de mudarse a la siguiente pareja.

3.3.2. Mínimos cuadrados Ortogonal

Dado que un *SLD* es no-lineal en sus parámetros ajustables, el algoritmo de retropropagación implementa un proceso de optimización no-lineal y puede ser atrapado en un mínimo local y converger lentamente. Se puede considerar fijar algunos parámetros de un *SLD*, tal que el resultado del sistema difuso es equivalente a una serie de expansión de algunas funciones básicas denominadas *funciones base difusas*. Esta expansión de funciones base difusas es lineal en sus parámetros ajustables; por lo tanto podemos usar el algoritmo clásico de *Gram-Schmidt* Mínimos cuadrados Ortogonal (*OLS* por su sigla en Inglés: Orthogonal Least Squares) para determinar las funciones base difusas

significativas y los parámetros restantes. El algoritmo *OLS* es un proceso de regresión de un paso y por lo tanto es más rápido que el algoritmo de retropropagación. Además el algoritmo *OLS* genera un *SLD* que no es sensible a inferir en sus entradas.

Una ventaja importante de usar funciones base difusas con respeto al uso de otras funciones bases como funciones polinómicas [5], es que una regla difusa lingüística *Si–entonces* es relacionada naturalmente a una función base difusa. Las reglas difusas lingüísticas *Si–entonces* siempre pueden ser obtenidas por expertos humanos que están familiarizados con el sistema bajo cierta consideración. Por ejemplo: los pilotos pueden describir propiedades de una nave aérea por reglas difusas lingüísticas *Si–entonces* y experimentados operadores de plantas de poder pueden proveer instrucciones operacionales en la forma de reglas difusas lingüísticas *Si–entonces*, etc. Estas reglas lingüísticas son muy importantes y siempre contienen información que no se tiene en las parejas de entrada-salida obtenidas al medir las salidas de un sistema para examinar entradas, porque las entradas para examinar quizás no son suficientemente ricas para excitar todos los modelos del sistema. Usando una expansión de funciones base difusas, fácilmente podemos combinar dos conjuntos de funciones base difusas, uno generado con parejas de entrada-salida usando algoritmo *OLS*, y otro obtenido de reglas difusas lingüísticas para una expansión de funciones base difusas singulares. Por lo tanto dicho conjunto es construido usando ambas informaciones numéricas y lingüísticas para una expansión uniforme.

Para formalizar estos conceptos, se presenta a continuación la definición de las funciones base difusas.

Considerando los *SLDs* dados en la ecuación 3.1, se definen las *funciones base difusas* (FBF por su sigla en Inglés) como

$$p_j(\underline{x}) = \frac{\prod_{i=1}^{n} \mu_{F_i^j}(x_i)}{\sum_{j=1}^{M} \prod_{i=1}^{n} \mu_{F_i^j}(x_i)}, \qquad j = 1, 2, \cdots, M \tag{3.10}$$

donde $\mu_{F_i^j}(x_i) = a_i^j \exp[-\frac{1}{2}(\frac{x_i - \bar{x}_i^j}{\sigma_i^j})^2]$ son funciones de membresía gaussiana. Los *SLDs*

2.11 son equivalentes a una expansión de FBF

$$f(\underline{x}) = \sum_{j=1}^{M} P_j(\underline{x})\theta_j \tag{3.11}$$

donde $\theta_j = \bar{y}^j \in R$ son constantes.

De 3.11 y 2.11 podemos ver que una FBF corresponde a una regla difusa *Si-entonces*. Especialmente, una FBF para una regla puede ser determinada de la siguiente forma: primero, se calcula el producto de todas las funciones de membresía para las terminologías lingüísticas en la parte de *Si* de las reglas, y se le llama la *seudo-FBF* para la regla; después de calcular las seudo-FBFs para todas las M reglas, la FBF para la j-ésima regla es determinada dividiendo la seudo-FBF para la j-ésima regla por la suma de todas las M seudo-FBFs. Una FBF puede ser determinada con base en una regla lingüística como la anterior, o con base en una pareja de entrada-salida numérica, como se describirá luego en el método de determinación de FBF inicial.

En la figura 3.2 se presenta un ejemplo de FBFs. Supongamos que tenemos cuatro reglas en la fórmula 2.11 con $\mu_{Fj} = \exp[-\frac{1}{2}(x - \bar{x}^j)^2]$, donde $\bar{x}^j = -3, -1, 1, 3$ para $j = 1, 2, 3, 4$, respectivamente (observe que las FBFs son determinadas nada más con base en los partes de *Si* de las reglas). Por lo tanto, $p_j = \frac{\exp[-\frac{1}{2}(x-\bar{x}^j)^2]}{\sum_{i=1}^{4}\exp[-\frac{1}{2}(x-\bar{x}^j)^2]}$, la cual es graficada en la figura 3.2 de izquierda a derecha para $j = 1, 2, 3, 4$, respectivamente. De la figura 3.2 podemos ver una propiedad interesante de las FBFs:

Las $p_i(x)'s$ y sus centros $\bar{x}^j$ que están dentro del intervalo [-3, 3] (el cual contiene todos los centros) se parecen a funciones gaussianas, mientras los centros de las $p_i(x)'s$ que están en los límites del intervalo [-3, 3] parecen funciones sigmoidales [7]. Es conocido en la literatura de redes neuronales que las funciones bases de gaussiana son buenas para caracterizar propiedades locales, mientras que las redes neuronales con funciones sigmoides no-lineales son buenas para caracterizar propiedades globales [23]. Estas FBFs parecen combinar las ventajas de ambas funciones gaussianas de base radial y redes neuronales sigmoides. Especialmente para regiones en el espacio de entradas U las cuales tienen puntos de muestra, siempre tomamos los puntos de muestra como centros de las FBFs, las FBFs cubren estos puntos con funciones parecidas a

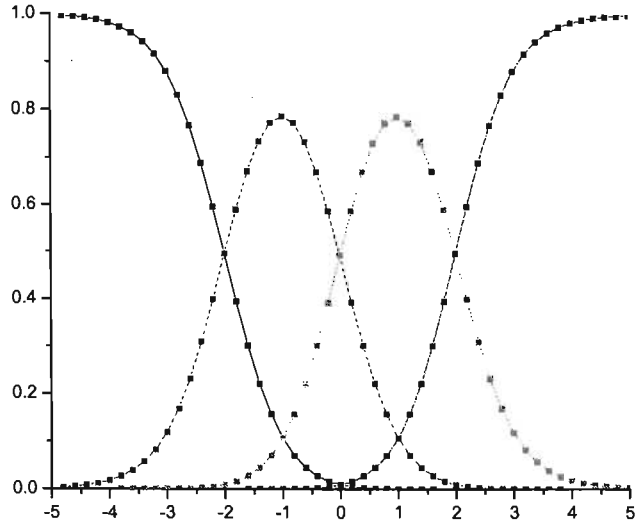

Figura 3.2: Un ejemplo de las FBFs

una gaussiana, entonces se puede obtener mejores resoluciones por la expansión de FBF sobre estas regiones. Por otro lado, en regiones de U las cuales no tienen puntos de muestra, las FBFs lo cubren con funciones parecidas a la de sigmoide, ya que fue mostrado que éstas tienen buenas propiedades globales [7, 23]. Por supuesto, todas las instancias previas son observaciones empíricas.

La ecuación (3.10) define un tipo de FBF, es decir que se define para los sistemas difusos con dedifusificador de promedio central, difusificador de singleton, inferencia de producto y función de membresía gaussiana. Los otros sistemas difusos pueden tener otras formas de FBFs. Por ejemplo, los sistemas difusos con inferencia mínima tienen una FBF en la forma (3.10) con operación producto reemplazada por operación mínima. Sin embargo, la idea básica permanece, es decir, examinar un sistema difuso como una combinación lineal de funciones, las cuales son definidas como FBFs. Diferentes FBFs tienen diferentes propiedades.

Podemos analizar la ecuación (3.11) desde dos puntos de vista. Primero, si enfocamos todos los parámetros $\alpha_i^j, \bar{x}_i^j$ y σ_i^j como parámetros de diseño libre, entonces la

expansión de FBF (3.11) es no-lineal en los parámetros. Para especificar tal expansión de FBF, tenemos que usar las técnicas de optimización no-lineal. Por ejemplo, usando el algoritmo de retropropagación que se mencionó antes. Por otro lado, podemos fijar todos los parámetros en $p_j\underline{x}$ en el primer momento del proceso de diseño de expansión de FBF, entonces el único parámetro de diseño libre es θ_j; en este caso $f(\underline{x})$ de (3.11) es lineal en los parámetros. Así adoptamos el segundo punto de vista. La ventaja de este punto de vista es que ahora podemos usar métodos muy eficientes de estimación de parámetros lineales, por ejemplo, el algoritmo de *Gram-Schmidt* Mínimo cuadrado Ortogonal [5] para diseñar las expansiones de FBFs.

Entrenamiento de mínimos cuadrados Ortogonal

Para describir cómo trabaja el algoritmo de aprendizaje de Mínimos cuadrados Ortogonal *(OLS)*, tomemos la expansión de funciones base difusas (3.11) como caso especial del modelo de regresión lineal

$$d(t) = \sum_{j=1}^{M} p_j(t)\theta_j + e(t) \tag{3.12}$$

donde $d(t)$ es la salida del sistema, θ_j son parámetros reales, $p_j(t)$ son conocidos como regresores que son funciones fijadas por las entradas $\underline{x}(t)$ del sistema, es decir,

$$p_j(t) = p_j(\underline{x}(t)) \tag{3.13}$$

y $e(t)$ es una señal de error correlacionada con los regresores. Supongamos que nos dan N parejas de entrada-salida: $(\underline{x}^0(t), d^0(t)), t = 1, 2, \cdots, N$. Nuestra tarea es diseñar una expansión de FBF $f(\underline{x})$ tal que el error entre $f(\underline{x}^0(t))$ y $d^0(t)$ sea minimizado.

Para presentar el algoritmo *OLS*, ordenamos (3.12) desde $t = 1$ a N en la siguiente forma de matriz:

$$\underline{d} = P\underline{\theta} + \underline{e} \tag{3.14}$$

donde $\underline{d} = [d(1), \cdots, d(N)]^T, P = [\underline{p}_1, \cdots, \underline{p}_M]$ con $\underline{p}_i = [p_i(1), \cdots, p_i(N)]^T$, $\underline{\theta} = [\theta_1, \cdots, \theta_M]^T$, y $\underline{e} = [e(1), \cdots, e(N)]^T$. El algoritmo de *OLS* transforma el conjunto de $\underline{p}_i$ a un conjunto de vectores básicos ortogonales, y solamente usa los vectores

básicos significantes para formar la expansión de FBF final. Para ejecutar el proceso *OLS*, primero necesitamos fijar los parámetros a_i^j, x_i^j, y σ_i^j en la FBF $p_j(\underline{x})$ con base en las parejas de entrada-salida.

Determinación de FBF inicial

Seleccionar N $p_j(\underline{x})$ iniciales de la forma (3.10) (para este caso, M en (3.10) equivale a N), con los parámetros determinados de la siguiente forma: $a_i^j = 1, \bar{x}_i^j = x_i^0(j)$, y $\sigma_i^j = [\max(x_j^0, j = 1, 2, \cdots, N) - \min(x_i^0(j), j = 1, 2, \cdots, N)]/M_s$, donde $i = 1, 2, \cdots, n, j = 1, 2, \cdots, N$, y M_s es el número de FBFs en la expansión de FBF final. Asumimos que M_s está dada con limitación práctica; en general, $M_s \ll N$.

Seleccionamos $a_i^j = 1$ porque $\mu_{F_i^j}(X_i)$ son funciones de membresía difusas las cuales pueden ser asumidas para llevar a cabo un único valor de membresía en algunos centros $\bar{x}_i^j$. Seleccionamos los centros $\bar{x}_i^j$ a ser puntos de entrada en las parejas entrada-salida dadas. Finalmente, la elección temprana de σ_i^j tendría que hacer que las FBFs uniformemente cubran la región de entrada cruzada por los puntos de entrada en las parejas dadas de entrada-salida.

A continuación, usaremos el algoritmo *OLS*, basado en el proceso clásico de *Gram-Schmidt* de ortogonalización para seleccionar las FBFs significativas desde las N FBFs determinadas por el método de determinación de FBF inicial.

En el primer paso, para $1 \leq i \leq N$, calcular

$$\underline{w}_1^{(i)} = \underline{p}_i, \qquad g_1^{(i)} = (\underline{w}_1^i)^T d^0 / ((\underline{w}_1^{(i)})^T \underline{w}_1^{(i)}) \tag{3.15}$$

$$[\text{err}]_1^{(i_1)} = (g_1^{(i)})^2 (\underline{w}_1^{(i)})^T \underline{w}_1^{(i)} / (\underline{d}^{0T} \underline{d}^0) \tag{3.16}$$

donde $\underline{p}_i = [p_i(\underline{x}^0(1)), ..., p_i(\underline{x}^0(N))]^T$, y $p_i(\underline{x}^0(t))$ son dadas por el método de determinación de FBF inicial. Buscar

$$[\text{err}]_1^{(i_1)} = \max[\text{err}]_1^{(i)}, (1 \leq i \leq N), \tag{3.17}$$

y seleccionar

$$\underline{w}_1 = \underline{w}_1^{(i_1)} = \underline{p}_{i_1}, \qquad g_1 = g_1^{(i_1)} \tag{3.18}$$

En el k-ésimo paso, donde $2 \leq k \leq M_s$, para $1 \leq i \leq N, i \neq i_1, ..., i \neq i_{k-1}$, calcular

$$\alpha_{jk}^{(i)} = \underline{w}_j^T \underline{p}_i / (\underline{w}_j^T \underline{w}_j), \qquad 1 \leq j \leq k$$

$$\underline{w}_k^{(i)} = \underline{p}_i - \sum_{j=1}^{k-1} \alpha_{jk}^{(i)} \underline{w}_j, \quad g_k^{(i)} = (\underline{w}_k^{(i)})^T \underline{d}^0 / ((\underline{w}_k^{(i)})^T \underline{w}_k^{(i)}) \tag{3.19}$$

$$[\mathrm{err}]_k^{(i)} = (g_k^{(i)})^2 (\underline{w}_k^{(i)})^T \underline{w}_k^{(i)} / (\underline{d}^{0T} \underline{d}^0) \tag{3.20}$$

buscar

$$[\mathrm{err}]_k^{(i_k)} = max[\mathrm{err}]_k^{(i)}, \qquad 1 \leq i \leq N, \quad i \neq i_1, ..., i \neq i_{k-1}), \tag{3.21}$$

y seleccionar

$$\underline{w}_k = \underline{w}_k^{(i_k)}, \qquad g_k = g_k^{(i_k)}, \tag{3.22}$$

resolver el sistema triangular

$$A^{(M_s)} \underline{\theta}^{(M_s)} = \underline{g}^{(M_s)}, \tag{3.23}$$

donde

$$A^{M_s} = \begin{bmatrix} 1 & \alpha_{12}^{i_2} & \alpha_{13}^{i_3} & \cdots & \alpha_{1M_s}^{i_{M_s}} \\ 0 & 1 & \alpha_{23}^{i_3} & \cdots & \alpha_{2M_s}^{i_{M_s}} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & \alpha_{M_s-1,M_s}^{i_{M_s}} \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{3.24}$$

$$\underline{g}^{(M_s)} = [g_1, ..., g_{M_s}]^T, \qquad \underline{\theta}^{(M_s)} = [\theta_1^{(M_s)}, ..., \theta_{M_s}^{(M_s)}]^T \tag{3.25}$$

La expansión de FBF final es

$$f(\underline{x}) = \sum_{j=1}^{M_s} p_{i_j}(\underline{x}) \theta_j^{(M_s)} \tag{3.26}$$

donde $p_{i_j}(\underline{x})$ son el subconjunto de las FBFs determinadas por el método de determinación de la FBF inicial con i_j determinado por los pasos previos.

Unos comentarios de este algoritmo *OLS* son los siguientes:

1. El propósito del algoritmo *OLS* original de *Gram-Schmidt* es ejecutar una descomposición para P, es decir, $P = WA$, donde W es un matriz ortogonal, y A es un matriz triangular-superior con elementos unitarios en la diagonal. Sustituir $P = WA$ a (2.47), tenemos que $\underline{d} = WA\underline{\theta} + \underline{e} = W\underline{g} + \underline{e}$, donde $\underline{g} = A\underline{\theta}$ tiene el mismo significado como lo usamos en este algoritmo *OLS*, y el $\alpha_{jk}^{(i)}$ corresponde a los elementos de A. Este algoritmo *OLS* no completa la descomposición de $P = WA$, pero selecciona algunas columnas del campo de P.

2. El $[\text{err}]_k^{(i)} = (g_k^{(i)})^2(\underline{w}_k^{(i)})^T\underline{w}_k^{(i)}/(\underline{d}^{0T}\underline{d}^0)$ representa la proporción de reducción de error debido a $\underline{w}_k^{(i)}$; por lo tanto este algoritmo selecciona FBFs significantes con base en su proporción de reducción de error. Que es, las FBFs con más grandes proporciones de reducción de error son retenidas en la expansión de FBF final.

En la literatura, podemos encontrar muchos algoritmos o esquemas para el entrenamiento de SLD[43], pero los algoritmos de retropropogación y *OLS* son las dos metodologías más populares.

3.3.3. Algoritmos genéticos

Diseñar un sistema adaptable basado en reglas difusas es equivalente a buscar una configuración de conjuntos difusos y/o reglas difusas, en este sentido puede ser observado como un problema de optimización. El criterio de optimización es el problema que tenemos y el espacio de búsqueda es el conjunto de parámetros que codifica las funciones de membresía y reglas difusas.

Hay dos tipos de métodos de optimización convencionales: Los directos (que no requieren más que los valores de la función objetivo), como **simplex**, y los indirectos, de ascenso (que, además requieren los valores de la o las derivadas de la función), como el de **Newton** y el descenso por máxima pendiente [28].

En años recientes, los algoritmos genéticos en particular y la computación evolutiva en general como un algoritmo de optimización llaman mucho la atención de los investigadores en diversas ramas de la computación y la ingeniería. Estos algoritmos

proporcionan una técnica de optimización universal que asimila el tipo de adaptación genética que ocurre en la evolución.

Los algoritmos genéticos son métodos heurísticos de búsqueda inspirados en lo que sabemos acerca del proceso de la evolución natural. El marco matemático fue desarrollado a finales de los 60´s, y es presentado en el libro de ingeniería de Holland [16].

El proceso básico de búsqueda AGs sigue el concepto del principio *Los más aptos sobreviven* de Darwin y mejora el fitness de las soluciones, generación por generación a través de un proceso de cuatro pasos: evaluación, reproducción, recombinación y mutación. El *fitness* es un índice de desempeño, el cual determina si una solución es buena o no. Por usar una población de soluciones procesales, los *AGs* pueden efectivamente explorar muchas regiones del espacio de búsqueda simultáneamente. Esta habilidad de búsqueda paralela y la propiedad libre de proceso de búsqueda son las razones principales de porqué los *AGs* son convenientes para búsquedas globales óptimas y robustas.

El alma de *AGs* es usar una regla simple para explotar un espacio complejo de respuesta en detalle e intentar de buscar una respuesta conveniente, si allí existe una. El beneficio de *AGs* es que cuando otros algoritmos no pueden resolver el problema, el algoritmo genético todavía tiene oportunidad de resolverlo.

Algoritmo genético simple (AGs)

Los *AGs* propuestos por Holland son llamados generalmente Algoritmos Genéticos Simples o *SGA* (por su sigla en Inglés). Aquí repasamos brevemente los pasos de *SGA* y su terminología antes de la discusión en los detalles de *AGs*.

En el principio de proceso de búsqueda de *AGs*, un conjunto inicial de soluciones llamado *población* es generado al azar. Cada individuo en la población se denomina *cromosoma*, el cual es una cadena compuesta de *genes* y representa parámetros codificados de la solución. Las cromosomas evolucionan por iteraciones sucesivas, llamadas *generaciones*. El primer paso en cada generación es decodificar y evaluar cromosomas actuales por una función predefinida de desempeño. La *progenie* (offspring) son los cromosomas nuevos de la siguiente generación y son formados por *padres* mediante

los operadores del reproducción, cruza y mutación. Los cromosomas de una generación nueva son decodificados y evaluados. Se repiten estos procesos para varias generaciones, entonces se converge a un cromosoma que tiene el mejor desempeño y por lo tanto la solución óptima o cercana a la óptima al problema se obtiene. El proceso de SGA es presentado en figura 3.3. En esta sección ilustramos los componentes básicos de *AGs*.

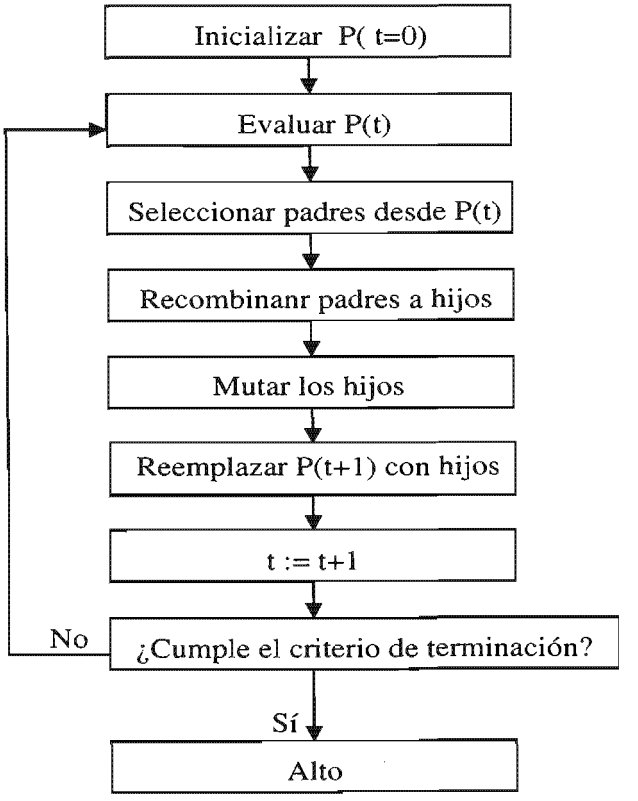

Figura 3.3: Estructura de un algoritmo genético

Fundamentos de *AGs*

En esta sección ilustramos los componentes básicos de los *AGs*. Hay muchos métodos para cada operador, aquí nada más representamos los detalles acerca del que usamos

en este trabajo, los otros se describen brevemente. El lector interesado puede consultar en la literatura.

Codificación y decodificación

El mecanismo de codificación que usa un método conveniente para representar las variables del problema de optimización es la base de la estructura de *AGs*. La representación más común es una cadena binaria, pero en el diseño de controlador difuso, la codificación no-binaria (dígito entero o real) es más conveniente; entonces se adopta esta codificación en este trabajo. El AG codificado en dígito real también ha sido mostrado como capaz de explorar el espacio grande de solución sin sacrificar la precisión o memoria además se converge más rápido que un AG con codificación binaria. Cada gen en los cromosomas representa un parámetro del controlador difuso, por supuesto cada cromosoma representa un controlador difuso. En el proceso de codificación, asignamos un individuo al azar (es decir un cromosoma), desde este cromosoma, conducimos los otros cromosomas de la población entre un rango predefinido de variación aleatoriamente, y el individuo debe ser decodificado a los parámetros de la función objetivo cuando se evalúa el desempeño.

Función de desempeño (fitness)

Para determinar si un cromosoma es una buena solución de nuestro problema o no, depende mucho de la *función de fitness*, para ser optimizada vía *AGs*. Cada cromosoma tiene su propio valor de fitness al evaluar la función de fitness. Los mejores cromosomas tienen más probabilidad a sobrevivir. Si la función es inadecuada para nuestro problema, tendríamos una solución pobre aunque el valor de fitness es alto. Los *AGs* trabajan bien nada más si le das una instrucción buena. Así pues es importante definir la función de fitness apropiadamente.

El mapeo de funciones objetivos a la forma de fitness

La manera en que los *AGs* resuelven nuestro problema es buscar un cromosoma que tiene máximo desempeño. Cuando el problema está en la forma de minimización, tenemos que transformarlo al problema de maximización. Una transformación sencilla se muestra en seguida [10]:

$$f(x) = \begin{cases} C_{max} - g(x) & cuando\ g(x) < C_{max} \\ 0 & otra\ situación. \end{cases} \tag{3.27}$$

donde *f(x)* es la función de fitness, *g(x)* es la función objetivo que necesita ser minimizada, y C_{max} es un constante que tiene muchas manera de escogerse.

Otra manera de hacer transformación es usar la función exponencial:

$$f(x) = \exp(-g(x)) \tag{3.28}$$

donde $g(x)$ es la función objetivo que necesita ser minimizada.

Escalamiento de fitness

En la etapa más temprana de búsqueda, podría existir uno o pocos cromosomas con grados de desempeño extraordinariamente altos; estos individuos tienen probabilidad más alta a ser seleccionados que los otros, entonces ellos pueden rechazar información importante contenida en otros individuos y conducir a una *convergencia prematura.* En las etapas posteriores, la diferencia de grados de fitness entre la mayoría de los cromosomas puede ser muy poca para converger efectivamente a una solución óptima. Para prevenir las desventajas mencionadas, se adopta un escalamiento lineal de valores de desempeño. Suponga que f es el desempeño original y f' es el desempeño escalado; el escalamiento lineal tiene la siguiente forma:

$$f' = af + b \tag{3.29}$$

Evaluación de la población

Por el principio de evolución de Darwin, individuos superiores a sus competidores obtienen mejor oportunidad a ascender sus genes a la siguiente generación. Al igual

que la naturaleza, es necesario que los *AGs* establezcan algún criterio para decidir cuáles individuos en una población son mejores, y cuáles no. La idea es calificar a cada individuo de cada generación de un AG usando una función objetivo, darle una calificación o para usar el término biológico, una medida de su *grado de fitness*, que es un número real no negativo tanto más grande cuando mejor sea la solución propuesta por dicho individuo.

Aquí hay que tener en cuenta el tiempo de ejecución. Como se evalúa una vez en cada individuo de cada generación, si un AG es ejecutado con una población de tamaño 100 durante 100 generaciones, la función es evaluada 10, 000 veces. Además puede darse el caso de que la función de evaluación no tenga una regla de correspondencia explícita; esto es, una expresión algebraica. Por otro lado, hay ocasiones que según el problema que tenga, puede ocurrir que se tenga varias funciones objetivos.

Reproducción

La reproducción, también es conocida como selección, es uno de los tres operadores básicos de los algoritmos genéticos. Este es un proceso en el cual las cadenas de individuos son copiadas conforme a sus valores de fitness y un algoritmo de selección.

Después de calificar los individuos de la generación, cada individuo tiene su calificación, el algoritmo asimila lo que tiene la naturaleza, selecciona a los individuos más calificados, mejor adaptados al medio, para que tengan mayor oportunidad de reproducción.

De esta manera se incrementa la probabilidad de tener individuos "buenos"(con alta calificación) en el futuro. Si de una determinada generación de individuos se seleccionaran sólo aquellos con una calificación mayor o igual que cierto número para pasarlos a la siguiente generación, obviamente que en esta generación la calificación promedio superará el promedio de la generación anterior. Esta selección ocasiona que haya más individuos buenos, explota el conocimiento que se ha obtenido hasta el momento, procurando elegir lo mejor que se haya encontrado, elevando así el nivel de

adaptación de toda la población.

Este método parece que es conveniente para que mejore la población y converja el algoritmo, es decir, que la población se acumule alrededor de un genotipo óptimo. Pero en realidad esto no es cierto, lo que ocurrirá es que la población se acumulará rápidamente alrededor de algún individuo que sea bueno, comparativamente con el resto de los individuos considerados a lo largo de la ejecución del algoritmo, pero este individuo puede no ser el mejor posible. A esto se la suele llamar convergencia prematura. No se puede asegurar, pero sí procurar que lo anterior no ocurra.

Además de la explotación, es necesario que exista exploración. Un AG debe, no sólo seleccionar de entre lo mejor que ha encontrado, sino procurar encontrar mejores individuos. A esto se dedican los operadores que serán descritos a continuación, los que aseguran que en todo momento exista cierto grado de variedad en la población, procurando con ello que no se "vicie"[28].

Hay muchas estrategias de selección, en general, se usa la estrategia de selección probabilística, tales como selección proporcional y selección torneo, selección ranqueo [28]. Aquí usamos la de torneo por su eficacia y su eficiencia computacional.

El primer paso de selección torneo es seleccionar dos o más (dos en nuestra caso) cromosomas de la última generación; después copiar el cromosoma que tiene desempeño más alto a la generación siguiente, este proceso se repite hasta que la generación nueva contenga tantos cromosomas como la anterior.

Cruzamiento

El operador cruzamiento es un operador de dominancia principal en los *AGs*. Se usa para producir descendientes que son diferentes de sus padres pero heredan la porción de material genético de sus propios padres. La cruza de los códigos genéticos de individuos exitosos favorece la aparición de nuevos individuos que hereden de sus antecesores características deseables.

Existen muchos mecanismos de cruzamiento el más común es el llamado *cruzamien-*

to de un punto: después de seleccionar dos individuos como padres al azar, se elige un punto de corte y se intercambian los segmentos análogos de las dos cadenas, conforman de dos cadenas nuevas como hijos. Además hay *cruzamiento de dos puntos*: se eligen dos puntos de corte y se intercambian los segmentos medios de ambas cadenas. Si se considera a los extremos de las cadenas como sitios continuos, como un anillo, entonces también se denomina *cruzamiento de anillo.* Y otro es el *cruzamiento uniforme*: para cada posición de bit de una cadena a generar se elige aleatoriamente el bit de la misma posición de alguna de las cadenas generadoras. Estos operadores se pueden ver en la figura 3.4.

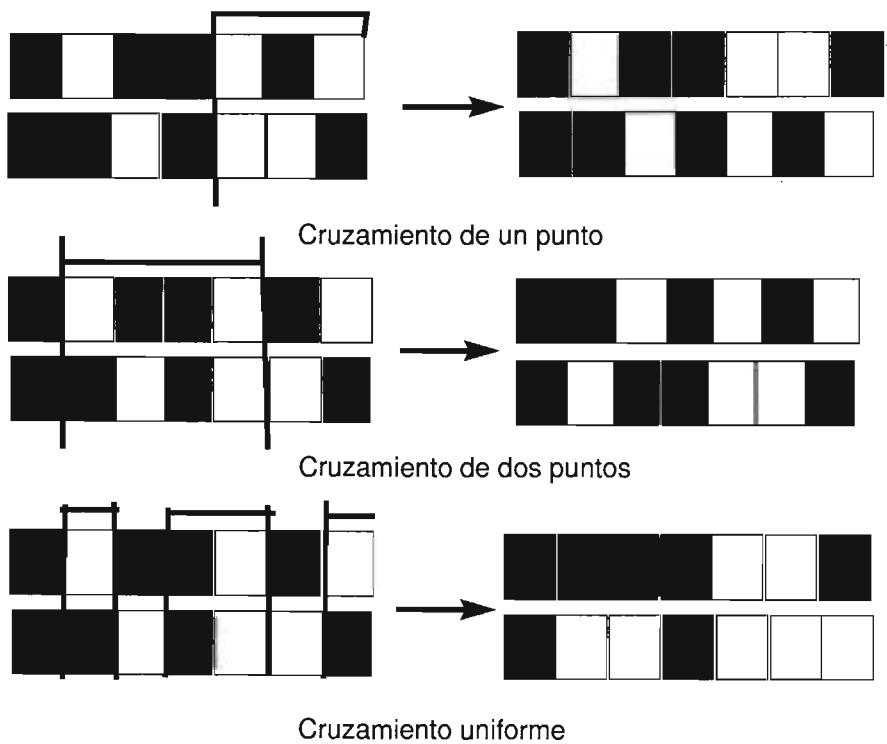

Figura 3.4: Tres tipos de cruzamiento

Pero en este trabajo, como usamos una codificación de dígito real, usamos una recombinación intermedia [3], que es un método de producir fenotipos nuevos alrededor

y entre los valores de fenotipos padres. Los descendientes son producidos según la regla:

$$O_1 = P_1 \times \alpha(P_2 - P_1) \tag{3.30}$$

donde α es un factor de escalamiento escogido uniformemente al azar en algún intervalo, típicamente [-0.25, 1.25] y P_1 y P_2 son cromosomas padres. Cada variable en los descendientes es el resultado de combinar las variables en los padres según la expresión anterior con una nueva α escogida para cada pareja de genes padres. En término geométrico, la recombinación intermedia es capaz de producir variables nuevas dentro de un hipercubo poco más grande que lo definido por los padres pero restringido por el rango de α. Al usar este operador, como se actúa en variables de decisión a sí mismos, tiene la ventaja de superar la limitación que después de la recombinación, los otros operadores con representación binaria al generar cambios que podrían no ser no expresados a los valores actuales de las variables de decisión, aunque los hijos sí pueden contener genes de ambos padres. Para que sea más intuitiva la manera de cruza, vea la figura 3.5:

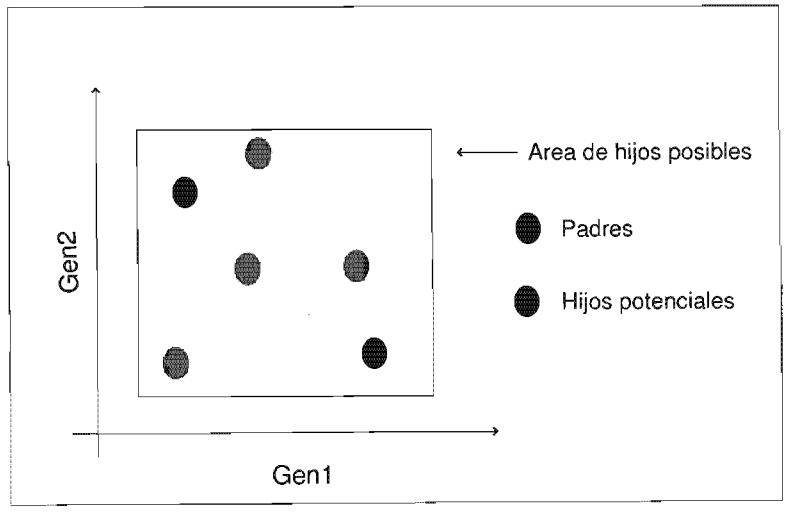

Figura 3.5: Efecto geométrico de cruzamiento intermedio

Mutación

Generalmente el operador mutación juega un papel secundario en el proceso genético. Pues los científicos descubrieron que algunas veces, muy pocas de hecho, el ADN-polimerasa (la enzima encargada de replicar el código genético) se equivoca y produce una mutación, una alteración accidental en el código genético de los seres vivos.

Ocasionalmente algunos elementos del código de ciertos individuos de un AG se alteran a propósito. Éstos se seleccionan aleatoriamente en lo que constituye el símil de una mutación. El objetivo es generar nuevos individuos, que exploren regiones del dominio del problema que probablemente no se han visitado aún. Esta exploración no presupone conocimiento alguno, no es sesgada. Aleatoriamente se buscan nuevas soluciones que quizás superen las encontradas hasta el momento. Ésta es una de las características que hacen aplicables los *AGs* a gran variedad de problemas: no presuponer conocimiento previo acerca del problema a resolver ni de su dominio, no sólo en la mutación sino en el proceso total. De hecho, el problema a resolver sólo lo determina la función de evaluación y la manera de codificar las soluciones posibles (la semántica de los códigos genéticos de los individuos). El resto de los subprocesos que constituyen el algoritmo son independientes y universalmente aplicables.

No obstante el hecho que la mutación puede tener un papel vital en un AG, debería notarse que ésta ocurre con un porcentaje pequeño de probabilidad, típicamente 0.001-0.01 de una población entera. Pero en algunas ocasiones, por ejemplo, cuando se codifica el genotipo usando número real o entero, en casos difíciles de mejorar el fitness con el operador de cruzamiento, se usa el operador de mutación, con un porcentaje alto.

En nuestro trabajo, no realizará la mutación ya que el operador de cruzamiento que usamos, cumplirá el objetivo de la búsqueda, además como no realiza este proceso, ahorrará el tiempo de cómputo.

Estrategia elitismo

La estrategia elitismo se realiza antes de las operaciones principales de *AGs*, tales

como reproducción, cruzamiento y mutación, en cada generación. Este proceso guarda los $n\%$ mejores individuos y pasan directamente a la siguiente generación sin procesar estas operaciones. La generación nueva es formada con los individuos guardados y los nuevos descendientes generados, así los mejores individuos no podrán ser destruidos por el proceso de evolución. Esta estrategia asegura que el máximo fitness aumentará continuamente en cada generación. El esquema de la estrategia elitismo es presentada en la figura 3.6.

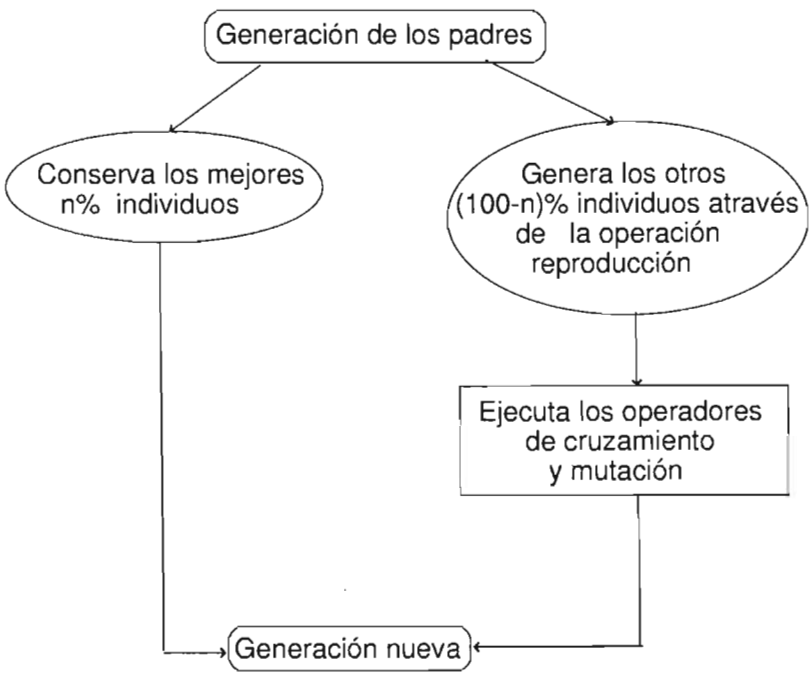

Figura 3.6: El esquema de la estrategia elitismo

En la literatura también se puede encontrar más operadores, tales como estrategia de extinción y de inmigración etc.

Parámetros

Los parámetros más comunes en los AGs son los siguientes:

1. El tamaño de la población.
2. El método de selección.

3. La tasa de reproducción.

4. El tipo de cruzamiento.

5. La tasa de cruzamiento.

6. La tasa de mutación.

7. La taza de elitismo.

La selección de operadores y de su porcentaje depende mucho del problema a resolver. Para algunos problemas, es muy sensible de la taza de los operadores. En este punto de vista, este también es un tema interesante de investigación.

3.4. Diseño de un sistema adaptable basado en la lógica difusa

Se ha mencionado que si un sistema basado en la lógica difusa equipado con un algoritmo de adaptación, se realizará la adaptabilidad. Aquí se propone usar los algoritmos genéticos para el entrenamiento, por lo que simplemente se denomina a la combinación como sistemas difusos genéticos (vea la figura 3.7). El objeto de un sistema difuso genético es automatizar el proceso de la adquisición de conocimiento en el diseño del sistema difuso. La tarea es realizada generalmente por una entrevista u observación de un experto humano. En esta sección se presentan unos métodos que son utilizados en la literatura y lo que se usa en este trabajo.

Con la realización de un AG se puede adaptar una parte o todos los componentes de la base de conocimiento del sistema difuso, como se puede ver en la figura 3.7. En este punto, es importante notar que una base de conocimiento difusa no es una estructura sino que está compuesta de bases de datos y bases de reglas. Por un lado, cada una juega un papel específico en el proceso de razonamiento difuso; por otro lado, en un sistema difuso, las funciones de membresía y conjuntos de reglas son codependientes. Por eso,

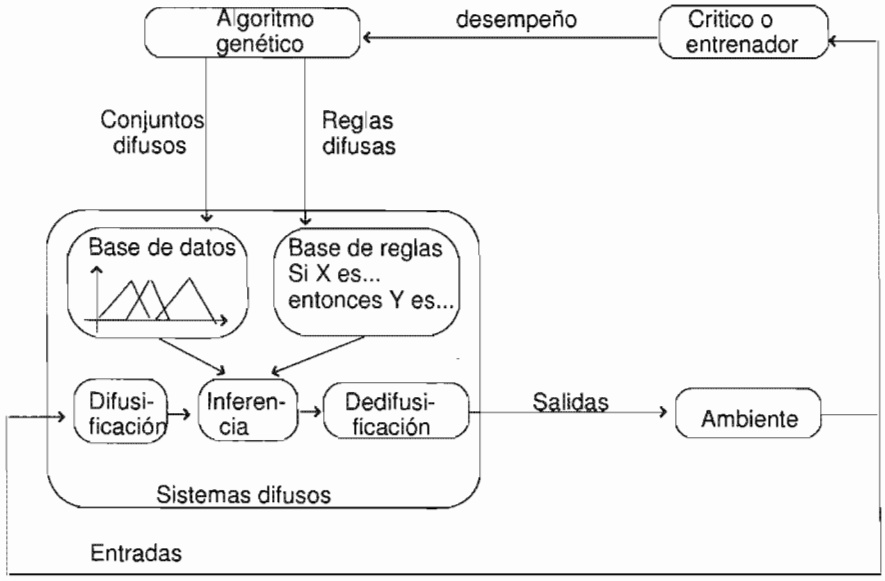

Figura 3.7: Sistema difuso genético.

los sistemas difusos genéticos son discriminados a lo largo de tres métodos principales: procesos de ajusto genético, procesos de aprendizaje genético y la combinación de los dos.

El primer método se concentra en optimizar el desempeño de un sistema difuso ya existente (es decir que la base de reglas ya está definida). El proceso de ajustamiento incluye la adaptación de la base de datos difusos, es decir parámetros de funciones de membresía. El segundo método se concentra en la derivación automática de reglas difusas en la base de reglas. Un proceso de aprendizaje difuso enfrenta una tarea mucho más difícil porque éste tiene que establecer una relación apropiada entre entradas y salidas al azar, no se optimiza el desempeño de un sistema que ya funciona al menos en forma aproximadamente correcta. El tercero diseña funciones de membresía y conjunto de reglas en una sola etapa.

A continuación se explicarán los métodos.

El Método 1. Ajuste de los parámetros.

Los conjuntos difusos son definidos por funciones de membresía, parametrizadas generalmente. El número de parámetros depende de su figura, por ejemplo, los conjuntos difusos triangulares son definidos por tres puntos característicos, los conjuntos difusos trapezoidales por cuatro puntos y los conjuntos difusos gaussianos por centro y ancho. El número de parámetros puede ser reducido si ciertas limitaciones son impuestas en la partición difusa. Muchos sistemas difusos emplean conjuntos difusos normalizados, que requieren el valor de membresía $\mu_{A_i}(x)$ de todos los conjuntos difusos A_i que suman la unidad

$$\sum_{i}^{l} \mu_{A_i}(x) = 1, \qquad \forall \ x \tag{3.31}$$

Entonces es suficiente definir los puntos centrales $c_1, \cdots, c_L$ de las funciones de membresía triangular para especificar la partición difusa entera de una variable. *F. Hoffmann* [15] mencionó una representación interesante la cual codifica las distancias Δ_i entre conjuntos difusos adyacentes sin considerar su locación absoluta

$$c_i = c_{i-1} + \Delta_i = c_0 + \sum_{k=1}^{i} \Delta_i \tag{3.32}$$

se muestra en la figura 3.8

El punto c_0 define el límite interior del rango de la variable. Debido a la limitación de la normalización como se puede ver en la figura 3.9, los puntos izquierdo l_i y derecho r_i de cada conjunto difuso A_i coinciden con los puntos centrales de los vecinos c_{i-1} y c_{i+1}. Este tipo de representación nada más necesita un parámetro Δ_i para cada conjunto difuso A_i.

2. El método 2 y 3: El proceso de aprendizaje.

X. Yao [48] mencionó otra representación de funciones de membresía. Usamos un ejemplo para explicar el método. Supongamos un sistema difuso que tiene cuatro variables de entrada y una variable de salida. Cada variable tiene cinco conjuntos difusos representando las descripciones ligüísticas: *muy lento, lento, medio, rápido, muy rápido.* Podemos usar los enteros 1-5 para representar estas cinco etiquetas, usar el entero 0 para representar la ausencia de una etiqueta, y usar un signo '-' para codificar "no". Por

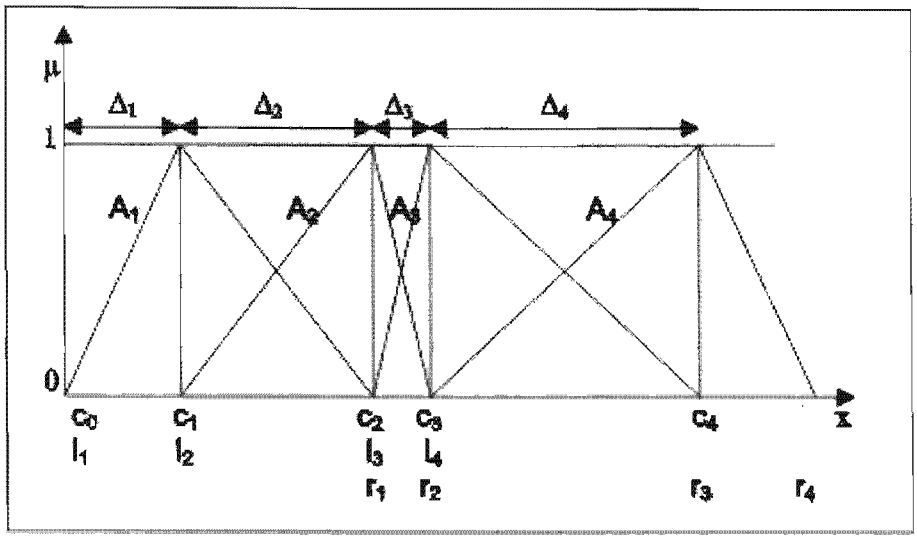

Figura 3.8: Representación genética de una partición difusa: el cromosoma codifica las distancias Δ_i entre los puntos centrales c_i de los conjuntos difusos adyacentes A_i y A_{i-1}.

ejemplo, -1 significa "no muy lento" cuando 1 significa "muy lento". En esta manera, una regla difusa puede ser completamente representada por 11 enteros. Por ejemplo, la regla: *Si entrada 1 es no lento, entrada 2 es no medio, y entrada 4 es rápido, entonces la salida es muy rápido.* puede ser codificada como -2-3045. (Nota: aquí se consideran reglas del tipo de *Mamdani*). Si el conjunto difuso incluye 20 reglas, entonces una cadena de enteros de longitud 100 puede representar el conjunto de reglas completamente.

Si se consideran los seis tipos de funciones de membresía que mencionamos en el capítulo 2 como candidatos, entonces pueden ser representadas por un entero de 1 a 6. Una función de membresía en este ejemplo es completamente determinada por tres valores: un punto inicial x_1, un punto terminal x_2 y el valor que asigna el tipo de función de membresía. Para tener un cromosoma homogéneo, los números enteros son seleccionados para representar el punto inicial x_1 y el punto terminal x_2 en lugar de valores reales. Si se supone que para la variable x cuyo rango dinámico es [a, b] y tiene n conjuntos difusos y si las funciones de membresía son uniformemente distribuidas sobre el rango superponiendo como se muestra en la figura 3.9, entonces los puntos

centrales $c_i (i = 1, 2, \cdots, n)$ de la i-ésima función de membresía son ubicados en

$$c_i = a + i * \text{paso} \qquad i = 1, \cdots, n, \quad \text{donde} \quad \text{paso} = \frac{b-a}{n+1}.$$

Limitamos el punto inicial x_1^i de la i-ésima función de membresía a variar únicamente entre c_{i-1} y c_i, y el punto terminal x_2^i de la i-ésima función de membresía puede variar entre c_i y c_{i+1}. Se supone un entero $s (s = 0, \cdots, 10)$ usado para representar x_1^i y x_2^i, entonces x_1^i y x_2^i pueden ser calculados desde el entero s usando la siguiente fórmula:

$$x_1^i = i * \text{paso} - \frac{\text{paso} * (10 + s)}{2 * 10} + a$$

$$x_2^i = i * \text{paso} + \frac{\text{paso} * (10 + s)}{2 * 10} + a, \qquad i = 1, \cdots, n.$$

Para un sistema difuso desconocido, generalmente no tenemos idea de cuántas reglas podrían ser incluidas en el conjunto de reglas antes de que el sistema sea diseñado. En este caso, un número máximo aceptable puede ser dado y el número de reglas difusas en el conjunto de reglas también podría ser evolucionado con esta limitante. Se supone que para el sistema mencionado el número máximo aceptable es 30, entonces la longitud total (en enteros) del cromosoma representado en el sistema es

$$1 + 5 * (5 * (2 + 1)) + 5 * 30 = 226$$

donde el primero 5 es 5 variables de entradas y salida, el segundo 5 es 5 conjuntos difusos que pertenecen a cada variable y el 30 es número máximo de reglas.

Y el sistema puede ser representado como

$$s_1 s_2 s_3 s_4 \cdots s_{14} s_{15} s_{16} s_{17} \cdots s_{76} s_{77} s_{78} s_{79} s_{80} s_{81} \cdots s_{222} s_{223} s_{224} s_{225} s_{226}$$

donde s_1 representa el número de reglas variando entre 1 y 30, s_2, s_3 representan el punto inicial y el punto final para el primer conjunto difuso de la primera variable de entrada y pueden variarse entre 0 y 10, s_4 representa el tipo de función de membresía para el primer conjunto difuso de la primera variable de entrada y pueden variarse entre 0 y 6, de s_5 a s_{76} codifican a las otras 24 funciones de membresía (punto inicial,

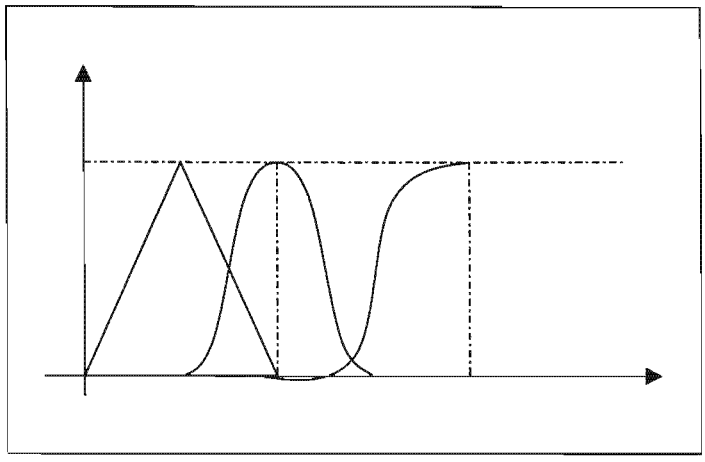

Figura 3.9: Tres funciones de membresía distribuidas uniformemente

punto final, el tipo de funciones de membresía. $24 * 3 = 72 = 76 - 5 + 1$), s_{77} a s_{81} representan la primera regla, y s_{222} a s_{226} representa la última regla posible. Debido a que s_1 especifica cuántas posibles reglas son codificadas en el cromosoma, solamente las primeras s_1 reglas son usadas para formar el conjunto de reglas, pero puede ocurrir que algunas de éstas no sean factibles, entonces se necesita verificar cada posible regla para ver si se representa una regla factible o no. Una regla que no tiene un antecedente o consecuente diferente de cero no es una regla factible y no puede ser incluida en el conjunto de reglas. Por ejemplo, suponga que tenemos una regla codificada como 12320. Su consecuente es cero, así no puede estar en el conjunto de reglas como una regla y el número de reglas factibles será $s_1 - 1$. Si todas las reglas posibles S_1 son infactibles (éste podría ocurrir en el principio de la ejecución del AG), entonces este cromosoma no contiene reglas factibles, no se puede formar un sistema difuso útil, y por lo tanto será asignado un valor aleatorio positivo pero muy pequeño como su valor de desempeño.

Función de desempeño

La selección de la función de desempeño es muy importante. La representación del genotipo codifica el problema en una cadena; sin embargo la función de fitness estima

el desempeño del sistema. Buscar una buena medida para la función de fitness para los sistemas que usan *AGs* es muy importante. No como los métodos tradicionales tales como los métodos basados en el gradiente, los *AGs* pueden ser usados para evolucionar cualquier tipo de funciones de fitness incluso aquellas que son no diferenciables, discontinuas, etc. Encontrar una buena medida de fitness puede ayudar a los *AGs* a desarrollar un sistema útil más fácil. Cómo se define la función de fitness para un sistema que necesita desarrollarse es dependiente del problema. Para problemas de precisión y estimación, una función usada muy común es una función relacionada con el error de mínimos cuadrados o error absoluto de diferencia.

$$E = \frac{1}{N}\sum_{i=1}^{N}(o_i - t_i)^2$$

$$E = \frac{1}{N}\sum_{i=1}^{N}|o_i - t_i|$$

donde N es el número de datos de entrenamiento, y o_i y t_i son la i-ésima salida obtenida y deseada, respectivamente. Suponer que el error máximo es $E_{\max}$, entonces una posible función de fitness es

$$F = E_{\max} - E.$$

Este tipo de funciones de error refleja el error absoluto; esto es, el error depende sólo de la diferencia entre las salidas obtenidas y deseadas. Por ejemplo, para una salida deseada de 10 y una salida obtenida de 9, y para una salida deseada de 2 y una salida obtenida de 1, el error absoluto es el mismo con valor de 1, y cada uno tendrá la misma contribución a la función de error. Pero 9 es una buena estimación de 10, mucho mejor que el 1 de 2. Usar las funciones de arriba para calcular el fitness, el sistema obtenido tendrá mejor precisión para una salida deseada grande que para la salida pequeña. Para tener sistemas que tienen precisión similar para cualquier salida deseada, las funciones de error relativo pueden ser introducidas, como unos ejemplos en el siguiente:

$$E = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{o_i - t_i}{t_i}\right)^2$$

$$E = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{o_i - t_i}{t_i}\right|.$$

Para problemas de clasificación, las funciones de error de arriba generalmente no son buenos candidatos. Porque si la clase **A** es mal clasificada como la clase **B**, obtendrán los mismos resultados cuando **A** es mal clasificada como cualquier otra clase. Normalmente, para problemas de clasificación, el número de clases mal clasificadas y el número de clases clasificadas correctamente son considerados en la función de error y función de fitness. Otros requisitos para sistemas también pueden codificarse en la función de fitness. Por ejemplo, si preferimos un sistema difuso con un número pequeño de reglas, entonces el número de reglas puede ser codificado en un factor de fitness normalmente por la sumatoria.

El uso de los operadores, cruzamiento, mutación, elitismo, etc., en los *AGs*, depende mucho del problema, por ejemplo, la mutación, en un problema representado en código binario o grey, lo muta cambiando de 0 por 1, o viceversa. Pero en el ejemplo usado aquí, el problema codificado en entero, entonces podría ser que mute el bit seleccionado para aumentar o disminuir 1 aleatoriamente dentro de su rango.

Dependiendo del problema que se va a resolver en la aplicación, en este trabajo se propone un algoritmo genético equipada con un sistema difuso en tipo TS que selecciona la función de membresía gaussiana por su suavidad, concisa notación y soporte infinito. Se codifica el problema en números reales, los individuos se forman con las constantes de los consecuentes de las reglas difusas, así ahorra el tiempo compucional de intercambio de los genotipos y los fenotipos. Además en este AG, a diferencia de los otros *AGs* propuestos de la literatura, donde se toma el cruzamiento de un punto o dos puntos, aquí se toma la recombinación intermedia la cual coincide con la codificación; pues con este método se converge más rápido para llegar a un criterio dado.

En la literatura se implementan muchos algoritmos genéticos para lograr el modelo de clasificación de sistemas difusos. En [33], *M. Setnes y H. Roubos* investigaron la complejidad y desempeño de la metodología, *P.J. Fleming y R.C.Purshouse* en [8] nos presentan un repaso acerca de los *AGs* en la ingeniería de sistemas de control, etc. Por otro lado, los sistemas difusos también pueden ser utilizados para optimizar los parámetros de algoritmos genéticos [48]. Además la lógica difusa también puede ser

usada en la codificación de algoritmos genéticos [35].

Capítulo 4

Aplicación al Control de Nivel de Líquido

4.1. Definición del problema

El control de nivel de líquido en tanques y el flujo entre tanques es un problema importante en el proceso industrial. El proceso industrial requiere que los líquidos sean bombeados, guardados en tanques y luego bombeados a otro tanque. Muchas veces los líquidos serán procesados por tratamiento químico o mezclando en los tanques, pero siempre se tiene que controlar el nivel de líquido en los tanques, así mismo el flujo entre los tanques. El problema de control de nivel y flujo de líquido en tanques está en el corazón de los procesos de ingeniería química. Claro sistemas de ingeniería química también están en el corazón de nuestra economía. Las industrias vitales en donde se realiza el control de nivel y flujo de líquidos son [26] :

* Industria de Petroquímica.
* Industria de la aplicación de papel.
* Industria de tratamiento de agua.

4.2. Sistema de tanques

Al saber la importancia de los sistemas de control de nivel de líquido, ahora veamos un sistema de dos tanques.

Un sistema de nivel de líquido en dos tanques.como el que se ilustra en la figura 4.1. tiene dos o más tanques conectados, por lo que también se llama sistema de tanques acoplados.

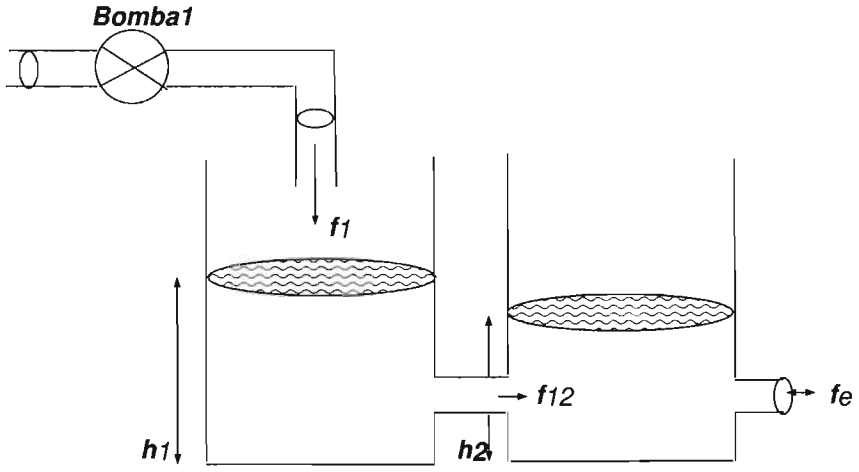

Figura 4.1: Sistema de dos tanques

La descripción del sistema

En este sistema, el líquido fluye al primer tanque por la bomba 1, una proporción f_i en cm^3, que corresponde a un actuator de voltaje en la implementación y controla el nivel de líquido del tanque 2; obviamente afecta la altura del líquido en el tanque 1 (denotada por h_1). Entre el tanque 1 y el tanque 2 hay una tubería conectando a los dos tanques, el líquido fluye saliendo del tanque 1 al tanque 2 por esta tubería en una proporción f_{12} afectando la altura h_1 del líquido en el tanque 1, y la altura h_2 del tanque 2. Después, el líquido fluye saliendo del tanque 2 en una proporción f_e por la salida. Con esta información, el objetivo es controlar el nivel de líquido h_2 en el segundo tanque, para que el error definido como $h_{2d} - h_2 \rightarrow 0$, ($h_{2d}$ es el nivel deseado).

Modelo matemático: Con el objetivo de simular el sistema adaptable en la computadora, se obtiene un modelo matemático del sistema de tanques. Los parámetros que se usan a continuación están definidos como:

$$\begin{aligned} az_i &: \text{ Coeficientes de flujo de salida de los tanques.} \\ h_i &: \text{ Niveles de líquido.} \\ f_{ij} &: \text{ Proporciones de flujos [m}^3\text{/sec].} \\ f_1 &: \text{ La proporción suministrada por el flujo [m}^3\text{/sec].} \\ A &: \text{ Sección transversal de lostanques [m}^2\text{].} \\ S_n &: \text{ Sección de la conexión de tubería [m}^2\text{].} \end{aligned} \tag{4.1}$$

en donde $i = 1, 2$; y $(i, j) = [(1, 2), (2, 0)]$.

Para el tanque 1, la ecuación de balance del flujo es

$$f_i - f_{12} = A\frac{dh_1}{dt} \tag{4.2}$$

para el tanque 2, la ecuación de balance del flujo es

$$f_{12} - f_e = A\frac{dh_2}{dt} \tag{4.3}$$

El modelo del sistema viene de dos balances de flujo y las ecuaciones para flujos. Dos balances de flujo pueden ser :

g : Aceleración ocasionada por la gravedad

$sgn(z)$: Signo del argumento z

$\triangle h$: Diferencia de nivel de líquido entre dos tanques conectados

az : Coeficientes de flujo de salida de los tanques (factor conectado, dimensiones, el rango de valores reales es de 0 a 1)

F : Proporción de flujo resultante en la tubería conectada por la ley de Turncelli

$$F = az\ S_n\ sgn(\triangle h)(2g|\triangle h|)1/2 \tag{4.4}$$

Entonces, el resultado es:

$$f_{12} = az_1\ S_n\ sgn(h_1 - h_2)(2g|h_1 - h_2|)^{1/2} \tag{4.5}$$

$$f_e = az_2 \; S_n \; sgn(h_2)(2gh_2)^{1/2} \tag{4.6}$$

Los valores numéricos de las constantes del sistema son tomados de el manual [1]:

Área seccional A: 0.0154 $[m^2]$

Área seccional S_n: $5 * 10^{-5}$ $[m^2]$

Altura máxima posible de líquido en los tanques h_{max}: $62[cm] \pm 1[cm]$

Flujo máximo q_1: $0{,}1[lt/s]$

Límites de voltaje de entrada al actuador: $\pm 15[v]$

El flujo entregado por la bomba se encuentra controlado por el voltaje, se muestra en la figura 4.2.

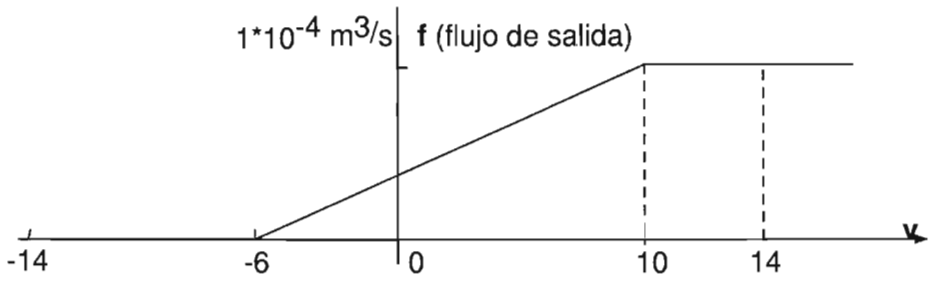

Figura 4.2: La función de flujo corresponde al voltaje

4.3. Diseño del controlador por un sistema adaptable

En esta sección se detalla el diseño de los distintos esquemas de control difuso utilizados: Sistema difuso del tipo Takagi-Sugeno, algoritmo genético.

Controlador utilizando un FIS[1] tipo Takagi-Sugeno

a. *Entradas*, ver la figura 4.3 :

- *Error en el tanque 2.* Definido como $e = h_{2d} - h_2$ es la señal de referencia o el punto de ajuste, h_2 es la altura del nivel del líquido en el tanque 2. Se asocian 3 conjuntos difusos en un intervalo de [-0.02, 0.02] [m], nombrando negativo, cero, positivo. Las funciones de membresía se consideran funciones de membresía gaussiana.

- *Derivada del error en el tanque 2.* se define como $\dot{e} = \frac{e(k)-e(k-1)}{T}$ donde T es el periodo de muestreo. Para esta entrada se consideran 3 conjuntos difusos: negativo, cero, positivo también, tomando un intervalo de [-0.0001, 0.0001] [m/s]. El tipo de función membresía también se considera gaussiana.

b. *Salida.* Aquí nada más hay una salida que es el voltaje Q de entrada a los actuadores que controlan la bomba. Corresponde a 9 reglas posibles, 9 constantes serán definidas en el intervalo [-6, 14](volt.).

Reglas

Las reglas pueden ser propuestas por experiencia. Como son pocas entradas, los antecedentes de las reglas toman todas las posibles combinaciones de dos entradas de error y de derivada de error (3 x 3 = 9). Entonces la superficie de control ahora es decidida por los consecuentes de las reglas.

Las reglas de control se plantearon de la forma canónica *si e es* A_i *y* $\dot{e}$ *es* B_j *entonces* Q *es* C_k, donde A_i se refiere a los conjuntos difusos de entrada para el error e y B_j a los conjuntos difusos de entrada de la derivada del error $\dot{e}$ y C_k a las constantes de la salida para cada regla.

Adelante se representa en la tabla (4.1) la superficie de control:

[1]Por su sigla inglés, Fuzzy Inference system

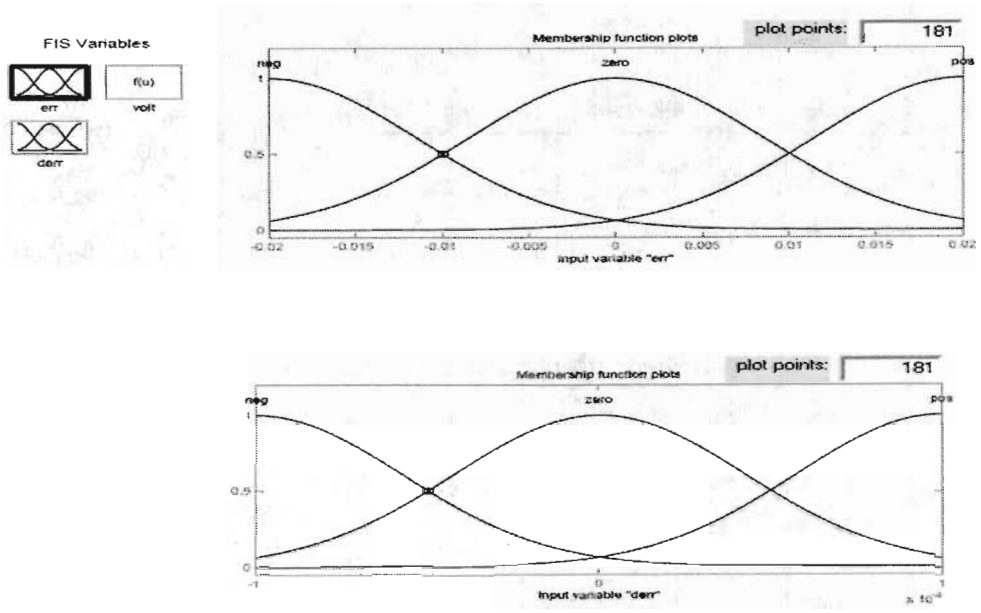

Figura 4.3: Las funciones de membresía de las entradas

		e		
		negativo	cero	positivo
$\dot{e}$	negativo	constante1	constante2	constante3
	cero	constante4	constante5	constante6
	positivo	constante7	constante8	constante9

Cuadro 4.1: Tabla de decisión

El algoritmo genético propuesto

En la sección 3.3.3 ya se presentó el algoritmo genético diseñado para optimizar el desempeño del controlador basado en lógica difusa, el cual es mejorado ajustando la parte del consecuente de las reglas.

Para este algoritmo, el número de individuos que se toma es 16. El número de generaciones al principio que se toma es 10, posteriormente se cambia dependiendo de los resultados obtenidos.

1) Para crear la población inicial, se define un FIS como patrón, desde este patrón se generan 15 FIS aleatoriamente, es decir los 16 individuos en números reales que representan los consecuentes de las reglas difusas para los procesos genéticos. Aquí se mide el rango de salida como un factor para generar individuos, multiplicado un número aleatorio en el rango [0, 1] a generar un parámetro para obtener un nuevo individuo.

2) Para la evaluación, aquí se propone una función de fitness [29] que evalúa el error entre la señal de referencia y la obtenida, evalúa las oscilaciones, evalúa el chattering así mismo el sobrepaso:

$$f = \exp(-\frac{a}{N}\sum(e_i{}^2 + \omega\Delta e_i{}^2) * i) \tag{4.7}$$

donde N es la duración de la simulación de evaluar el diseño; a es un número positivo usado para escalar el máximo fitness arriba o abajo; i es el índice de tiempo en la simulación, e_i es el error entre la señal medida y la señal deseada en el paso de la simulación i; Δe_i es el cambio de error en el paso de la simulación i; y ω es una constante no negativa usada para pesar entre el error y el cambio del error.

La definición de la función de fitness produce un valor entre 0 y 1, con valores más grandes corresponde a los controladores con mejor desempeño. La multiplicación del índice de tiempo i en la función de fitness se usa para asegurar errores pequeños de estado estable y a prevenir oscilaciones creciendo al infinito durante del diseño. Esta forma de evaluación también elimina la necesidad de incluir las condiciones de estabilidad asintótica, existencia, la posibilidad de alcanzar la meta del diseño o convergencia en el índice de desempeño [29].

Aquí lo definimos $a = 1$ y la función de fitness mencionada se calcula afuera de la simulación.

3) Se toman los 10 % mejores individuos de cada generación para llevar a cabo el *elitismo.*

4) Para el operador genético "Cruza", después de guardar los dos mejores individuos, se usa el algoritmo de selección de torneo para seleccionar dos individuos a realizar recombinación intermedia.

Con el AG diseñado se realiza una optimización de parámetros de reglas difusas mejorando el comportamiento de controladores.

4.4. Resultados de simulación

Los experimentos y resultados de la simulación:

La simulación se construyó en MATLAB/SIMULINK y el toolbox de lógica difusa. El objetivo de la simulación es verificar los resultados teóricos y dar las indicaciones para mejorar y automatizar el diseño del controlador. Por eso se utilizaron los mismos valores de los parámetros que los reales dentro de la simulación. Solo en el caso de las constantes de flujo a_1 y a_0 se tomaron valores constantes y no se realizó alguna modificación durante la simulación; esto es diferente de lo que sucede en la realidad. En este caso se tomaron los valores:

$$a_1 = 0{,}43 \qquad a_0 = 1{,}09$$

El diagrama de simulación se presenta en la figura 4.4 y los experimentos se describen abajo:

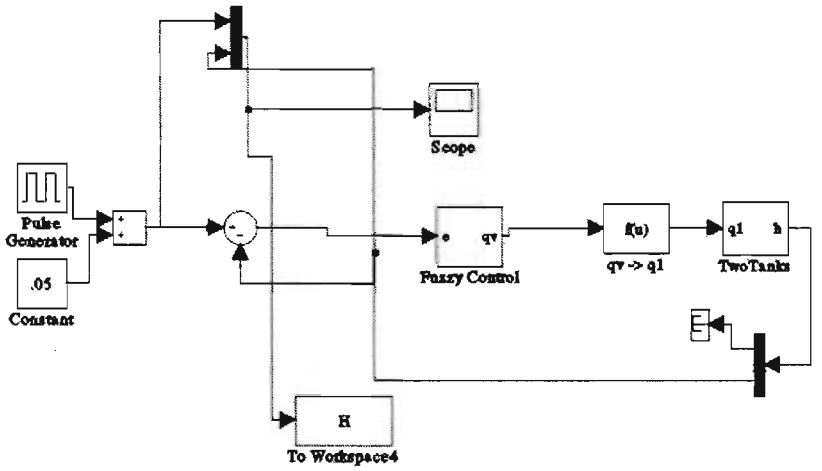

Figura 4.4: La diagrama de simulación

Para observar el funcionamiento del controlador, se ajusta el tiempo de adaptación, es decir, el número generaciones. Por los resultados obtenidos, podemos ver que en la figura 4.5 que aunque tiene una población pequeña, en la generación 30, ya tenemos controladores aceptables.

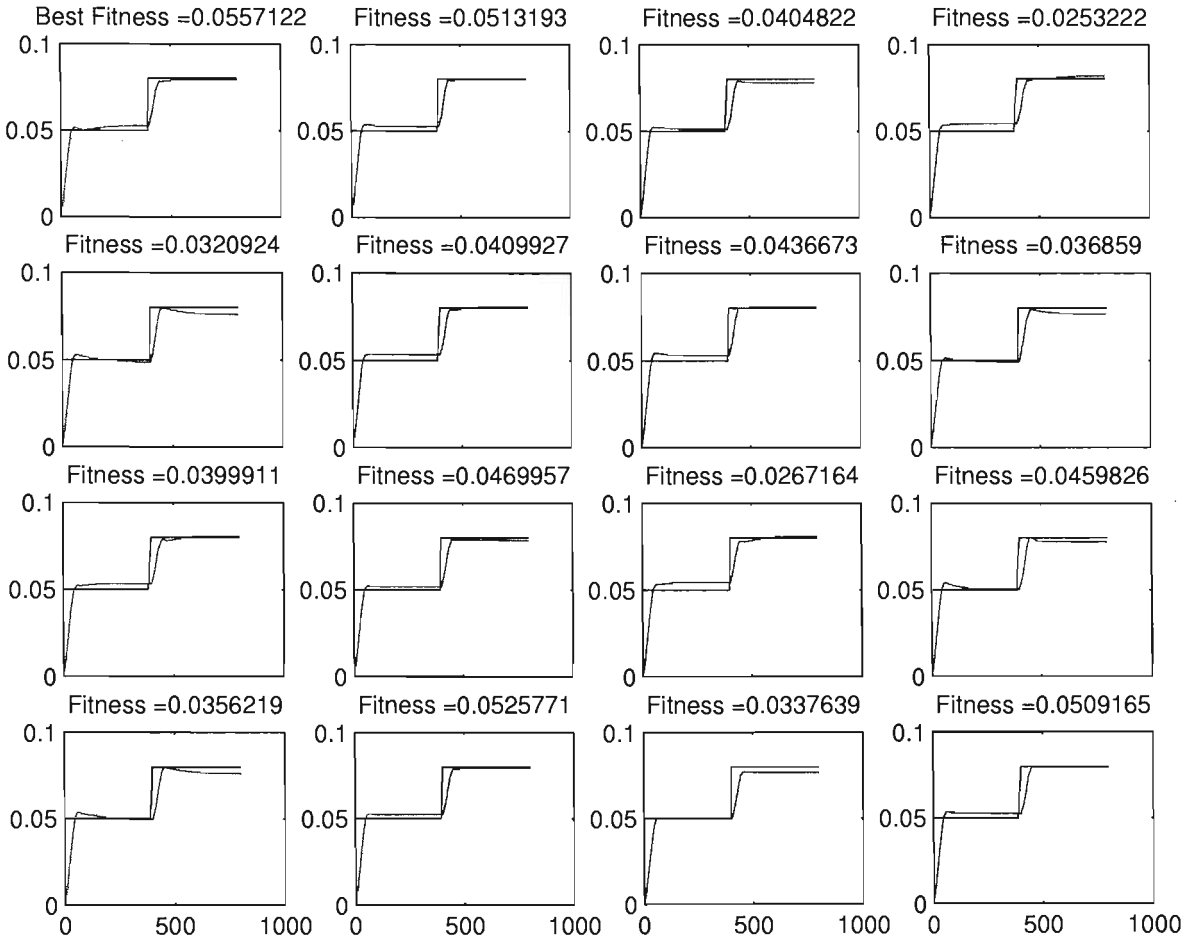

Figura 4.5: Los controladores obtenidos en la generación 30

Para mejorar el desempeño, pues 30 generaciones son muy pocas, lo aumentamos a 100 generaciones. En la figura 4.6 podemos ver todos los individuos de esta generación son aceptables, además el mejor que se ilustra en la figura 4.7 tiene un desempeño muy adecuado. En la figura 4.8 una presentación del cambio de fitness se ilustra que todavía estaba mejorando.

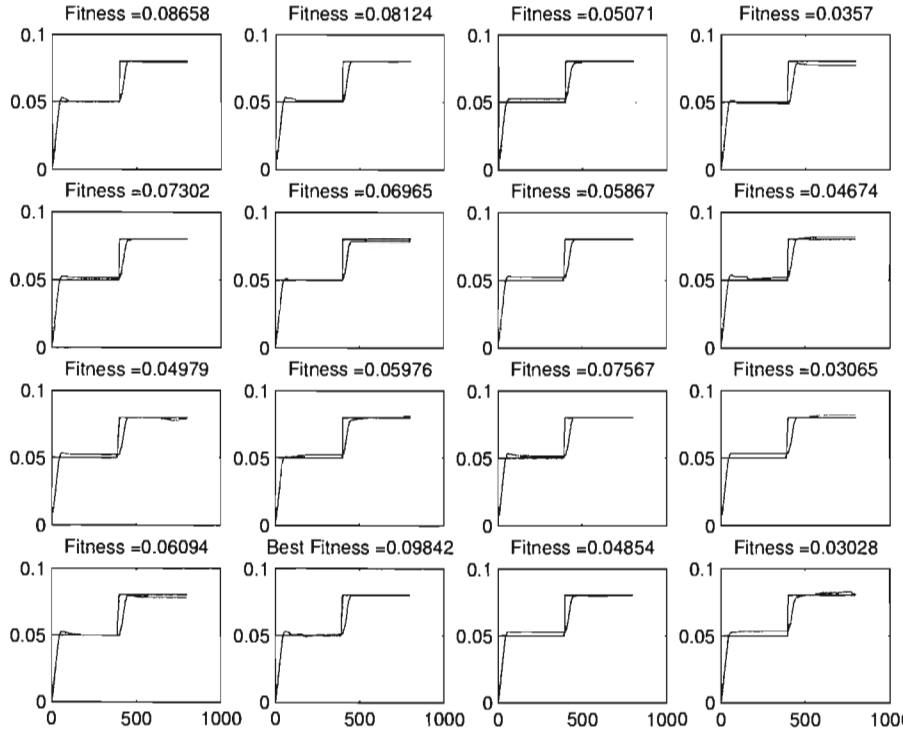

Figura 4.6: Los controladores difusos obtenidos en la generación 100

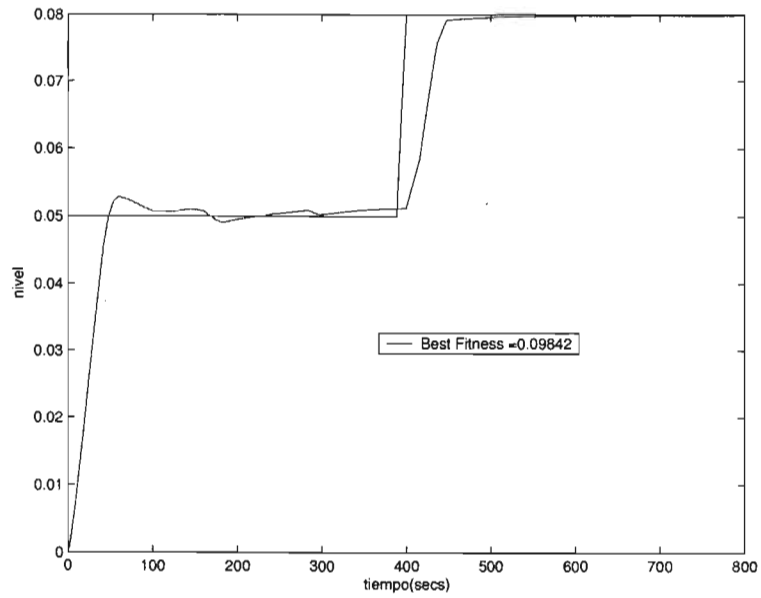

Figura 4.7: El óptimo de la generación 100

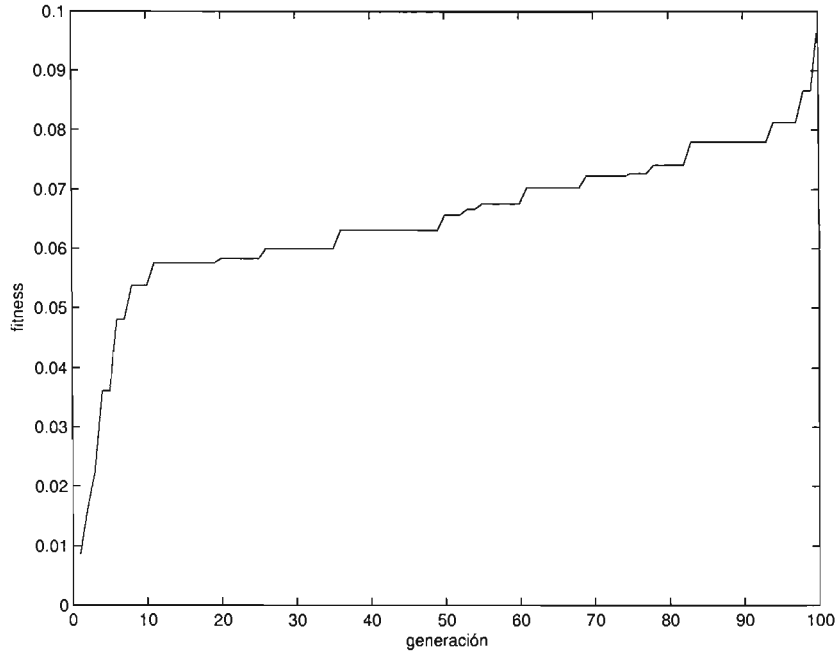

Figura 4.8: La presentación del cambio de fitness durante 100 generaciones

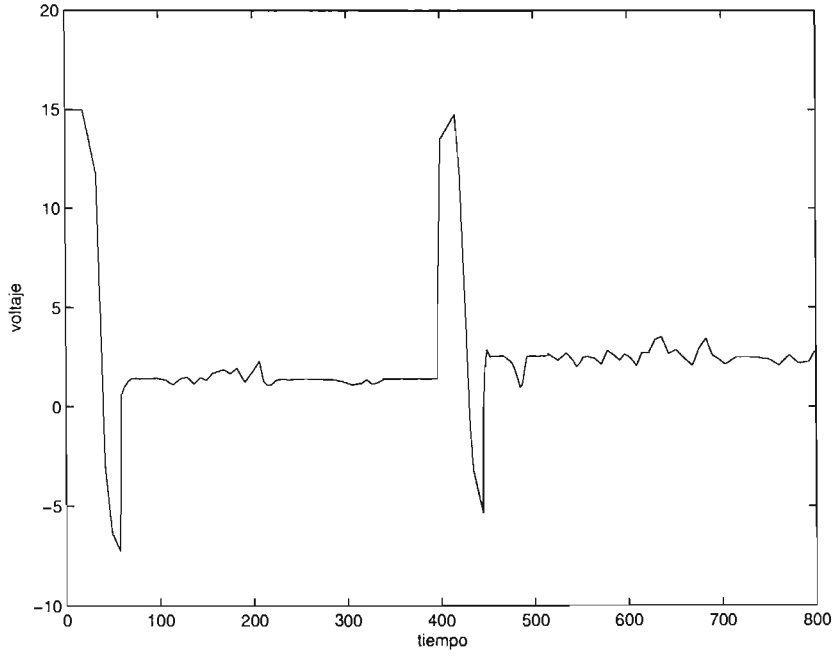

Figura 4.9: El voltaje obtenido en la generación 100

Si seguimos aumentando el número de generaciones, podríamos obtener mejores resultados todavía, teóricamente. Pero el resultado obtenido en la generación 200, como se pueden ver en las figuras 4.10,4.11,4.12, presenta una diferencia muy pequeña con lo que obtenemos en la generación 100. Es decir, cuando el desempeño llega un nivel de grado que está cercano del resultado óptimo, ya no es conveniente que aumente el tiempo de adaptación.

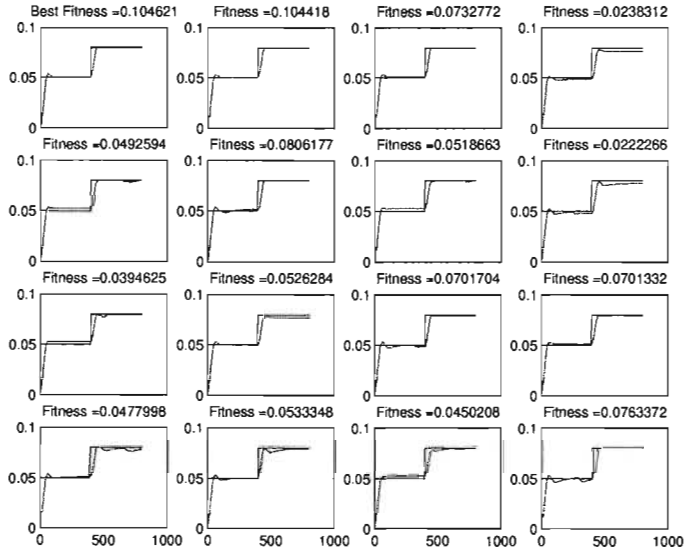

Figura 4.10: Los controladores obtenidos en la generación 200

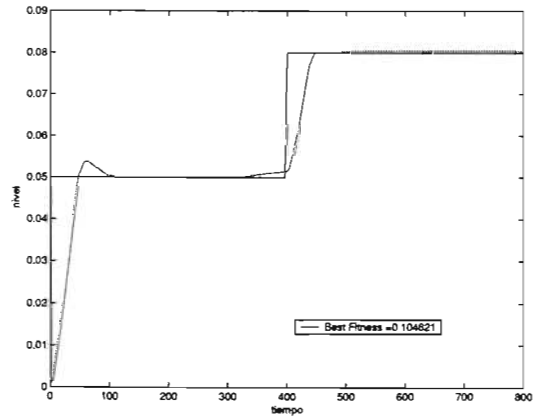

Figura 4.11: El óptimo de la generación 200

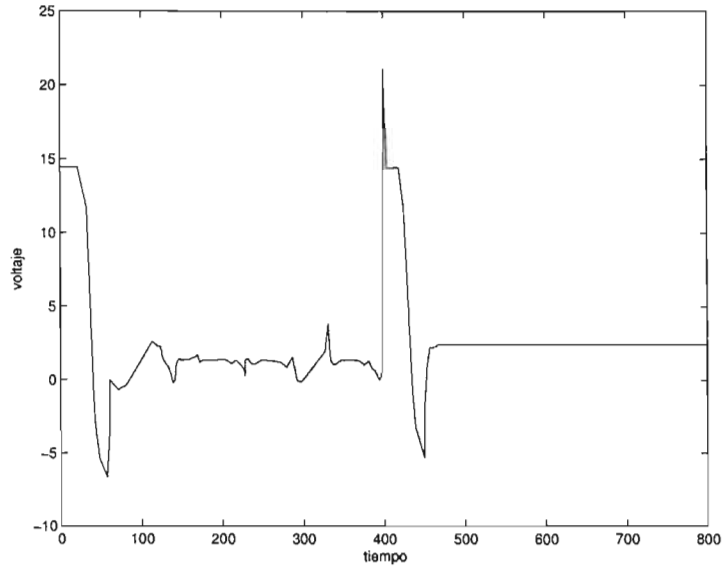

Figura 4.12: El voltaje en la generación 200

La figura 4.13, se presenta el superficie original de control, en la figura 4.14 ya se puede ver un superficie mucho mejor después de evolucionarse. En la figura 4.15, 4.16 son las figuras que muestran los cambios de las constantes de reglas difusas durante el tiempo de evolución.

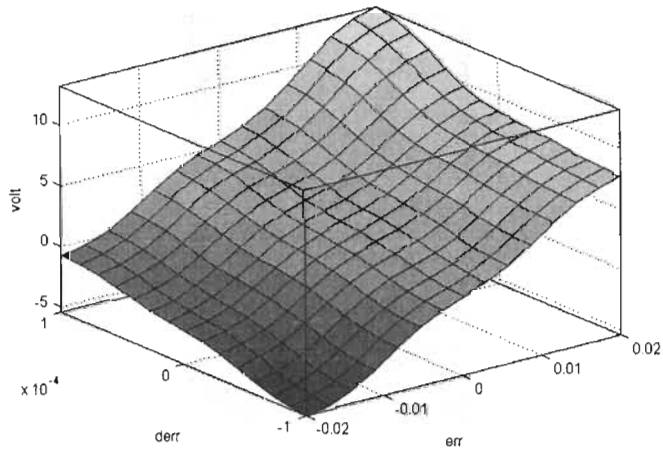

Figura 4.13: El superficie inicial del sistema

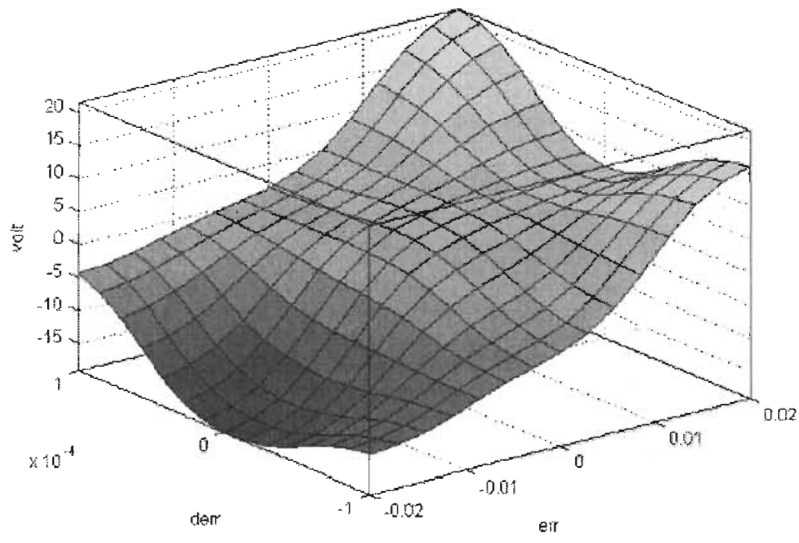

Figura 4.14: El superficie en generación 200

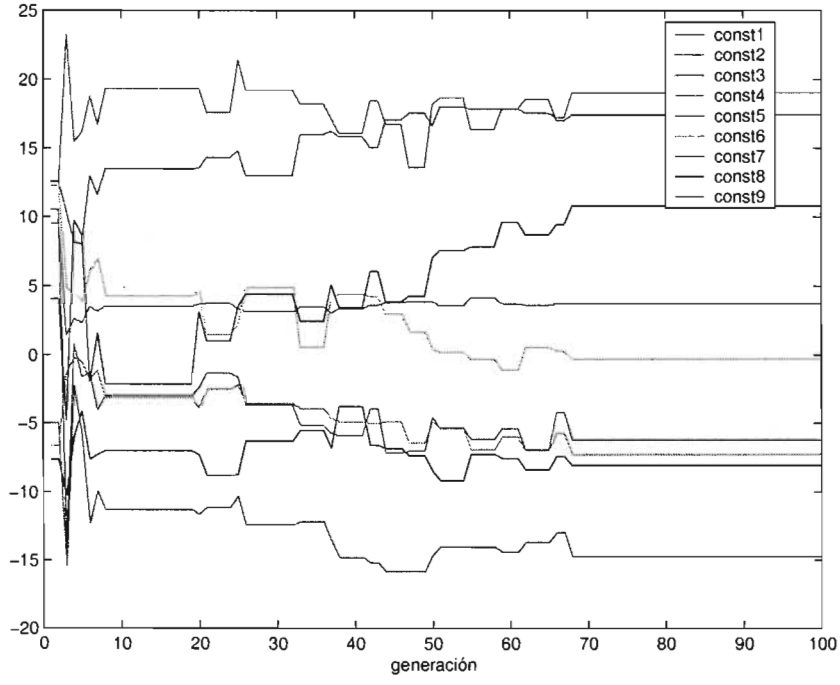

Figura 4.15: El cambio de los 9 consecuentes de las reglas difusas durante las 100 generaciones

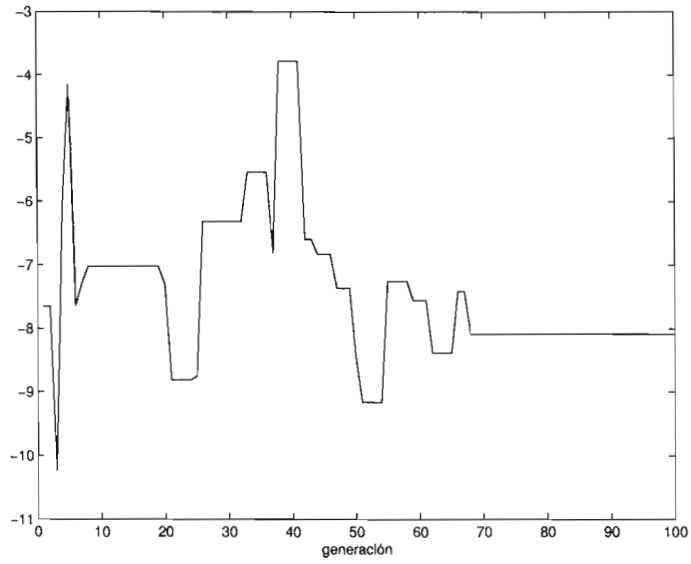

Figura 4.16: El cambio del consecuente de la primera regla durante las 100 generaciones

A continuación se muestra una tabla que presenta los resultados obtenidos de las salidas. Por el proceso estocástico del AG, los desempeños obtenidos no todos son semejantes, pero todos son aceptables.

	Generación			
	original	100_1	100_2	200
Voltaje	-6	-1.8308	-13.3961	-12.9304
	-3	-15.3724	-22.1044	-21.1653
	-1	-3.9014	4.6601	-3.1396
	1	-10.1948	-6.8136	-6.3954
	3	2.7226	4.2409	6.0084
	5.5	3.8379	-0.6918	0.1859
	8	12.9719	17.3853	17.9138
	9	29.0977	14.9357	4.5871
	14	8.5340	13.3021	23.9068

Cuadro 4.2: Tabla de las salidas del sistema

Capítulo 5

Conclusiones

En este trabajo se discute una metodología para diseñar sistemas adaptables basado en la lógica difusa es discutida, La metodología se basa en dos técnicas avanzadas: la lógica difusa y los algoritmos genéticos. En la aplicación de control de nivel de líquido se toma el controlador de tipo TS por su simplicidad en las reglas. Para mejorar el desempeño se propone un algoritmo genético de dígitos reales que optimiza simultáneamente los parámetros de los consecuentes de las reglas. También se muestra que la simplificación y la optimización pueden ser considerada para reducir la complejidad del modelo.

La metodología propuesta ha sido aplicada a un controlador difuso en este trabajo. Con el algoritmo genético propuesto se adapta la parte consecuente de las reglas que son los constantes en este caso y se han obtenido controladores con desempeño muy adecuado Los resultados muestran que no es necesario un número elevado de iteraciones para llegar a soluciones aceptables. Incluso para poblaciones relativamente pequeñas, se observó que el algoritmo propone soluciones que representan controladores que son adecuados. Como el problema de control no tiene muchas variables, no realiza la adaptación de reglas. Pero en caso de haber muchas variables, como las reglas se incrementan exponencialmente, ajustar el número de reglas sería una tarea muy importante. Se puede resolver por un AG con el método que se discute en el capítulo 3.

Con este trabajo se puede concluir que los sistemas adaptables basados en la lógica difusa, aunque han sido estudiado ya desde hace décadas, al combinar con técnicas avanzadas, todavía tienen un atractivo futuro. Además, esta metodología tiene facilidad de transferirse a la industria, porque la naturaleza de su paradigma es atractiva en la práctica de la ingeniería, y tiene capacidad de ser adaptados para encontrar la solución a problemas difíciles de la industria.

5.1. Observación

Como trabajo futuro se puede intentar ajustar toda la base de conocimiento de los *SLDs* para llegar a un nivel de diseño más automatizado, lo que implica que el diseñador no restringa las posibilidades.

Además, como el porcentaje de los operadores, tales como cruzamiento y mutación de *AGs* sí son muy sensibles y afectan el desempeño y la velocidad de convergencia de los individuos en el proceso de la evolución. Se puede considerar usar un *SLD* para ajustar los porcentajes de cruza y mutación para incrementar la adaptabilidad del sistema.

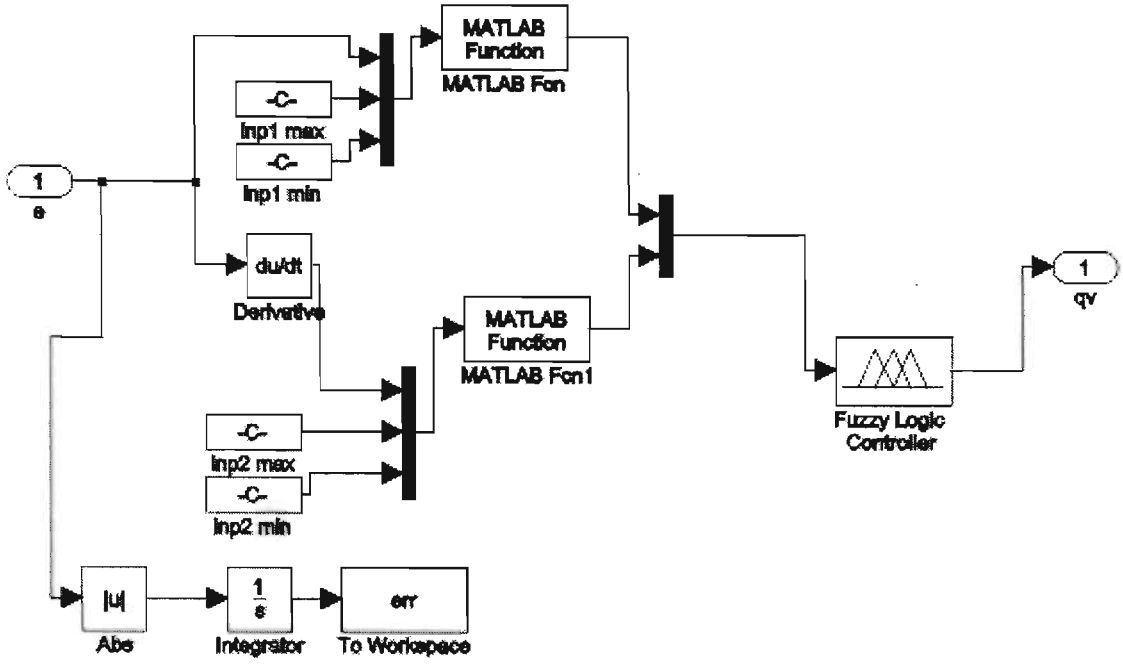

Figura 1: Modelo del controlador difuso

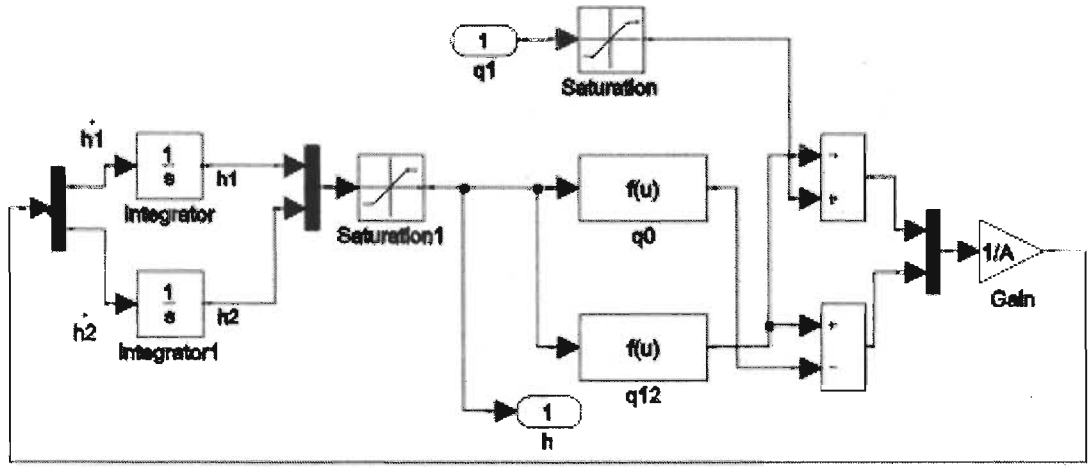

Figura 2: El modelo de 2tanques

Bibliografía

[1] Amira, DTS200 Laboratory Setup Three - Tanks - System, *Copyright amira GmbH 1993*, printed 21. December 1993.

[2] L. Ardissono, L. Console, I.A. Torre, An adaptive system for the personalized access to news, *AI Communications* **14** (3): 129-147 2001.

[3] J.E.Baker, Reducing bias and inefficiency in the selection algorithm, *Proc. IGGA* 2, 14-21, 1987.

[4] A. Bonarini, Evolutionary learning of fuzzy rules: Competition and cooperation In *Fuzzy modeling: Paradigms and Practice*, W.Pedrycz, ED. Norwell, MA: Kluwer, 265-284 (1996).

[5] S. Chen, S.A. Billingd, and W. Luo, Orthogonal least squares learning algorithm to non-linear system identification, *Int.J. Contr.* **50**(5), 1873-1896 (1989).

[6] S. Chen, C.F.N. Cowan, and P. M. Grant, Orthogonal least Squares learning algorithm for radial basis function network, *IEEE Transaction on Neural Networks* **2**(2), 302-309 (1991).

[7] G. Cybenko, Aproximation by superposition's of a sigmoidal function, *Mathematics of Control, Signals, and Systems* **2**, 303-314 (1989).

[8] P.J. Fleming, R.C. Purshouse, Evolutionary algorithms in control sustems engineering: a survey, *Control Engineering Practice* **10**, 1223-1241 (2002).

[9] I. G. French, C. K. S. Ho and C. S. Cox, Genetic algorithms in controller Structure selection, *Genetic algorithms Engineering Systems: Innovations and Applications* 12-14 September 1995, Conference Publication No. 414, IEE, 1995.

[10] D.E.Goldberg, *Genetic Algorithms in Serch, Optimization, and Machine Learning*, Mass.:Addison-Wesley, 1989.

[11] S.Haykin, *Neural Networks*, Macmillan (1994).

[12] F.Herrera, O.Cordón, and M. Lozano, On the combination of fuzzy logic and evolutionary computation: A short review and bibliography, in *Fuzzy Evolutionary Computation*, W. Pedrycz, ED. Norwell, MA: Kluwer, 57-77 (1997).

[13] F. Herrera, M. Lozano, and J.L. Verdegay, A learning process for fuzzy control rules using genetic algorithm, *Fuzzy Sets System* **100**(1-3), 143-158 (1998).

[14] F. Hoffmann, Soft computing techniques for the design of mobile robot behaviors, *J. Inform. Sci.* **122**, 241-258, Feberury 2000.

[15] F. Hoffmann, Evolutionary Algorithms for Fuzzy Control system Design, in *Proceedings of the IEEE* **89**(9), September 2001.

[16] J.H.Holland, *Adaptation in Natural and Artifical Systems*, Ann Arbor: The university of Michigan Press, 1975.

[17] S. M. Karim, A. C. Ricardo *et al.*, *Controladores adaptables basados en mecanismos de Inferencia Difusa.*

[18] C.Karr, Genetic algorithms for fuzzy controllers, *AI Expert* **6**, 26-33, Feberury 1991.

[19] B. Klein, A.R. Dengel, A. Fordan, SmartFIX: An adaptive system for document analysis and understanding, *Lecrure Notes in Computer Science* **2956**: 166-186, 2004

[20] A.V. Larichev, P.V. Ivanov, N.G. Iroshnikov et al, Adaptive system for eye-fundus imaging, *Quantum Electronics* **32** (10): 902-908, October 2002

[21] M.A.Lee, Automatic design and adaptation of fuzzy systems and genetic algorithms using soft computing techniques, Ph. D. dissertation, Univ. California, Berkeley, 1994.

[22] T.M. Lenton, M. van Oijen, Gaia as a complex adaptive system, *Philosophical Transactions of the Royal Society of London Series B-Biological Sciences* **357** (1421): 683-695, May 29 2002

[23] R. Lippmann, A critical overview of neural network pattern classifier, *Procedding of IEEE Workshop on Neural Networks for Signal Processing*, Princeton (N. J.), 266-275 (1991).

[24] A. Lofti Zadeh, Fuzzy sets, *Inf. Control***8**, 338-353 (1965).

[25] L. Magdalena and F. Monasterio, A fuzzy logic controller with learning through the evolution of its knowledge base, *Int. J. Approx. Reason.* **16**, 335-358 (1997).

[26] E.H.Mamdani, Application for fuzzy algorithms for the control of a dynamic plant, *Proc. IEEE* **121**, 1585-1588 (1974).

[27] D.L. Meredith, C.L. Karr, and K. Krishna Kamur, The use of genetic algorithms in the design of fuzzy logic controllers, 3rd Workshop on Neural Networks WNN'92, 549-545 (1992).

[28] A.K.Morales, J.G.Casas, *Algoritmos Genéticos* IPN, UNAM, FCE, 2002.

[29] K.C.Ng, Y.Li, D.J.Murria-Smith and K.C. Sharman, Genetic algorithms Applied to Fuzzy Sliding Mode Controller Design, *Genetic Algorithms Engineering Systems: Innovations and Applications* 12-14 September 1995, Conference Publication No. 414, IEE, 1995.

[30] D. Park, A. Kandel, and G. Langholz, Genetic-based new fuzzy reasoning models with application to fuzzy control, *IEEE Trans. Syst., Man, Cybern.* **24**, 39-47, January 1994.

[31] D. E. Rumelhart and J. L. McCleland, eds., *Parallel Distributed Processing I.II.* Cambridge, MA: MIT Press (1986).

[32] K. Saitwal, M.R. Azimi-Sadjadi, D.A. Reinke, Multichannel temporally adaptive system for continuous cloud classification from satellite imagery, *IEEE Transactions on Geoscience and Remote Sensing* **41** (5): 1098-1104 Part 2, May 2003

[33] M. Setnes and H. Roubos, GA-Fuzzy Modeling and Clasification: Comlexity and Performance, *IEEE Transactions on Fuzzy Systems***8** (5), October 2000.

[34] Y. H. Shi and Y. B. Chen, Implementation of Evolutionary Fuzzy systems, *IEEE Transactions on Fuzzy Systems* **7**(2), Abril 1999.

[35] S. K. Sharma, Fuzzy Coding of Genetic Algorithms, *IEEE Transactions on Fuzzy Systems* **7** (4), August 2003.

[36] Y. H. Shi, Implementation of Evolutionary Fuzzy Systems, *IEEE Transactions on Fuzzy Systems* **7**(2), April 1999.

[37] J. Shing and R. Jang, ANFIS: Adaptive -Network-Based Fuzzy Inference System, *IEEE Transations on Systems, Man, and Cybernetics* **23**(3), May/June 1993.

[38] H. Takagi and I. Hayashi, NN-Driven Fuzzy Reasoning, *Int. J. Approximate Reasoning* **5**, 191-21 (1991).

[39] T.Takagi and M.Sugeno, Derivation of fuzzy control rules from human operators control actions in *Proc.IFAC Symp. Fuzzy Inform., Knowledge representation and Decision Analysis* 55-60, July 1983.

[40] T. Terano, K. Asai and M. Sugeno, *Fuzzy Systems Theory and its Applications*, Academic Press, INC., 1991. Chaps. 1 and 2.

[41] M. Togai, and S. Chiu, A fuzzy accelerator for a programming environment for real-time fuzzy control, *Proc. 2nd IFSA Confress*, Tokyo, Japan, 147-151 (1987).

[42] M. Togai, and H. Watanabe, Expert system on a chip: an engine for real-time approximate reasoning, *IEEE Expert Syst. Mag.* **1**, 55-62 (1986).

[43] L. X. Wang, *Adaptive fuzzy systems and control: Design and stability analysis*, PTR prentice Hall, 1994.

[44] L. X. Wang, Stable Adaptive Fuzzy Controllers with Application to Inverted Pendulum Tracking, *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics* **26**(5), October 1996.

[45] L. X. Wang, Design and Analysis of Fuzzy Identifiers of Nonlinear Dynamic Systems, *IEEE Transactions on Automatic Control* **40**(2), January 1995.

[46] S. Wang, Generating fuzzy membership functions: A monotonic neural network model, *Fuzzy Sets and Systems* **61**, 71-81 (1994).

[47] N.Watanabe, Statistical Methods for Estimating Membership Functions, *Japanese Journal of Fuzzy Theory and Systems* **5**(4), (1979).

[48] X. Yao, Evolving Artificial Neural Networks, in *Proceeding of IEEE* **87**(9), September 1999.

[49] http://www.k1.inf.tu-dresden.de/ fritzke/Fuzzy/paper/t.html.

[50] http://wwww.aptronix.com/fide/whyfuzzy.htm.

[51] http://necsi.org/guide/concepts/adaptive.html, Yaneer Bar-Yam.

[52] http://www.gaianet.fsbusiness.co.uk/gaiatheory.html