Capítulo 5

Aprendizaje de reglas con modificadores semánticos

5.1 Modelos difusos descriptivos y aproximativos

Uno de los principales problemas con los que nos encontramos cuando utilizamos un algoritmo de aprendizaje consiste en establecer una discretización correcta de las variables que se mueven sobre referenciales continuos. Cuando esta discretización de los referenciales continuos de las variables se realiza mediante conjuntos difusos, diremos que el algoritmo de aprendizaje trabaja con modelos difusos.

Con respecto a la forma en la que se construyen los dominios de estas variables, podemos considerar dos líneas de actuación diferentes en los algoritmos de aprendizaje sobre modelos difusos:

i) aquellos que requieren que los dominios de las variables estén previamente definidos, es decir, que se conozca a priori el conjunto discreto de valores que puede tomar una variable, y

ii) aquellos que durante el proceso de aprendizaje establecen el dominio de las variables.

Los primeros son conocidos como modelos difusos descriptivos (MDD), ya que el conocimiento que se extrae de un determinado sistema se expresa exclusivamente en términos de los valores fijados a los dominios previamente a la ejecución del proceso de aprendizaje. Los segundos, son conocidos como modelos difusos aproximativos (MDA), donde el algoritmo de aprendizaje tiene dos tareas: encontrar los valores apropiados que deben tomar las variables y en función de estos valores extraer el conocimiento que permita describir al sistema.

Los MDD trabajan sobre espacios de búsqueda finitos, lo que conlleva que la capacidad de representación del sistema que se desea aprender esté limitada por los dominios definidos sobre las variables. Además, si se desea que el conocimiento extraído de un sistema sea fácilmente interpretable por una persona que trabaje con dicho sistema, es necesario que los dominios sean definidos por un experto. Por el contrario, los MDA no requieren la participación de un experto, y en este sentido, los algoritmos son más autónomos. Esta automonía del experto provoca que el conocimiento extraído sea difícil de interpretar desde un punto de vista humano, en la mayoría de los casos. Además, los MDA trabajan sobre espacios de búsqueda que en principio son infinitos, lo cuál requiere un esfuerzo mayor durante el proceso de aprendizaje.

En base a las características anteriores, podemos considerar que los MDD son apropiados para problemas donde se requiere que el conocimiento obtenido sea fácilmente interpretable, mientras que los MDA son apropiados para problemas donde lo fundamental es encontrar una buena representación del comportamiento del sistema, aún a pesar de que el resultado sea difícil de interpretar.

Una de las premisas que inicialmente se estableció en el desarrollo del sistema de aprendizaje SLAVE, consistía en que el conocimiento extraído por él, debía ser fácilmente interpretable desde un punto de vista humano, y por esta razón, como hemos visto en capítulos anteriores, se han introducido en su diseño factores que fuerzan al algoritmo a obtener este tipo de conocimiento. En este sentido, SLAVE es un MDD.

El hecho de que SLAVE sea un MDD, implica que tiene una fuerte dependencia de las discretizaciones propuestas por el experto para las variables que intervienen en el problema. Con los siguientes ejemplos, ilustramos esta dependencia.

Ejemplo 5.1 *Supongamos que se desea aprender un sistema que tiene una única variable antecedente X que sabemos que se mueve en el rango [0,3] y una variable consecuente Y que puede tomar dos valores $\{ClaseA, ClaseB\}$. Además, conocemos que el sistema tiene el siguiente comportamiento:*

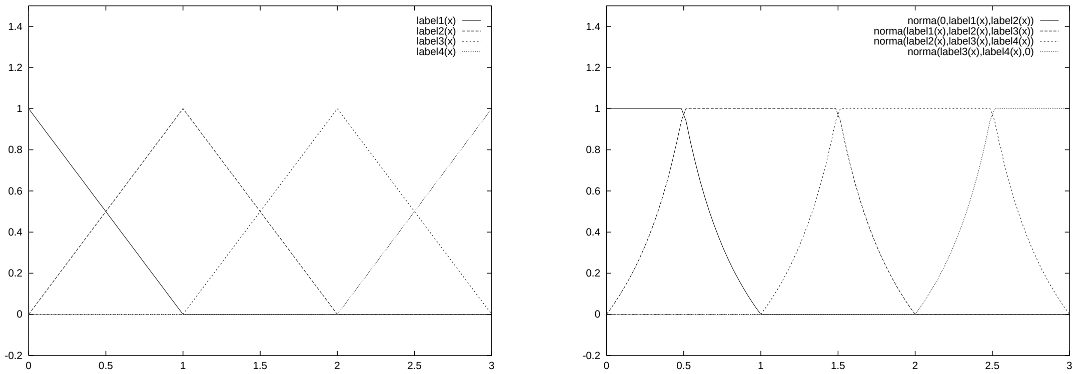
Problema 1: *SI $X \in [0.5,1.5]$ ENTONCES Y es Clase-A*
EN OTRO CASO Y es Clase-B

Para el aprendizaje de dicho sistema, el experto propone la discretización sobre el referencial de la variable X utilizando cuatro etiquetas difusas triangulares, como muestra la figura 5.1(a).

La figura 5.1(b) muestra la alteración de las etiquetas del dominio al aplicar la función de adaptación U de SLAVE

Como solución, SLAVE devuelve las dos reglas que muestran las figuras 5.2(a) y 5.2(b) respectivamente.

En la figura 5.2(c) podemos observar como las dos reglas discriminan sobre un conjunto de ejemplos generados a partir de la discretización propuesta para la variable. Por tanto,



(a) Dominio inicial de la variable X (b) Dominio modificado por la función U

Figura 5.1: Dominio de la variable X

SLAVE consigue una representación correcta del comportamiento del sistema, en base al dominio definido por el experto.

Sin embargo, una inadecuada discretización de las variables continuas conlleva en general un incremento del error en la capacidad de predicción del algoritmo de aprendizaje. Para ilustrar este problema proponemos el siguiente ejemplo.

Ejemplo 5.2 *Supongamos que se desea aprender el comportamiento de los siguientes dos problemas:*

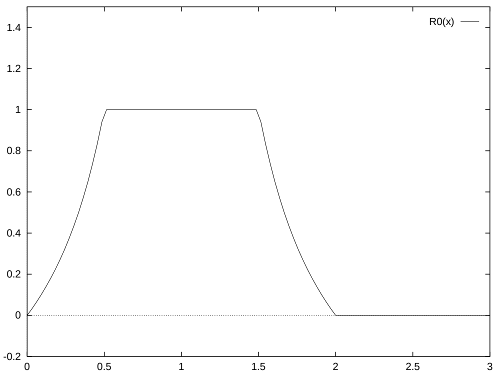
Problema 2: *SI $X \in$ [0.7,1.3] ENTONCES Y es Clase-A*
EN OTRO CASO Y es Clase-B

Problema 3: *SI $X \in$ [0.9,1.8] ENTONCES Y es Clase-A*
EN OTRO CASO Y es Clase-B

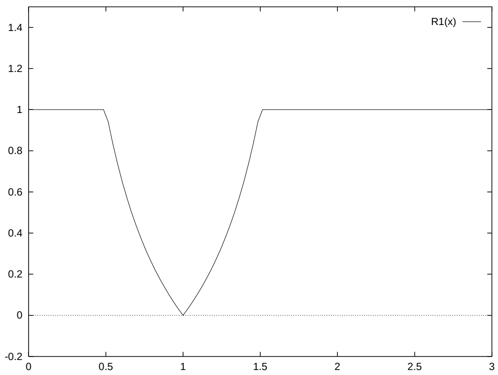
donde ambos tienen una variable predictiva $X \in$ [0,3] y una variable de clasificación Y. El dominio definido sobre la variable X para los dos problemas es igual que el propuesto para el ejemplo 5.1 y se muestra en la figura 5.1 y los valores que puede tomar la variable consecuente Y, son al igual que en el ejemplo anterior dos $\{ClaseA, ClaseB\}$.

Aplicando SLAVE a ambos problemas, obtenemos el mismo conjunto de reglas, que además coinciden con el conjunto de reglas que se obtuvieron para el problema del ejemplo anterior.

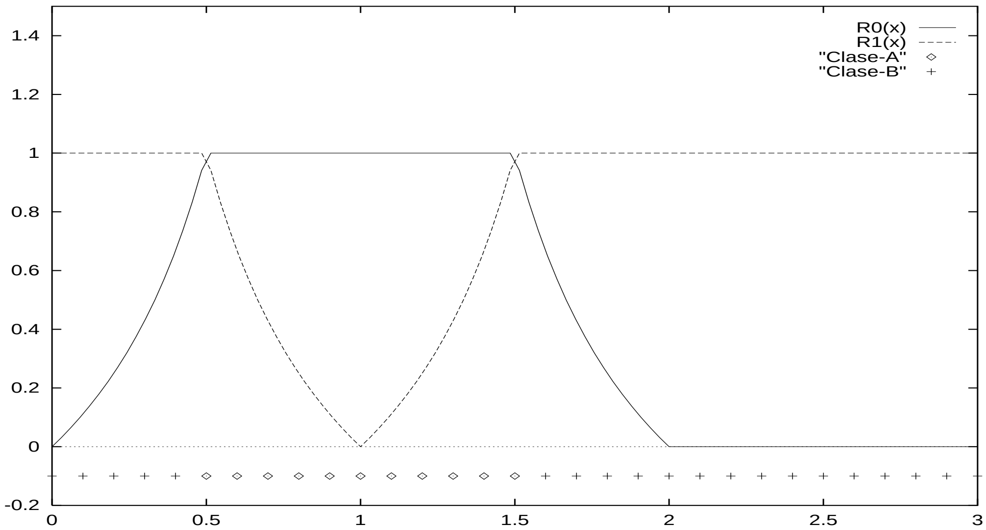
Como podemos observar en la figura 5.3(a) y en la figura 5.3(b), las reglas obtenidas no permiten realizar una correcta clasificación de los ejemplos que se le proponen. En concreto, en ambos casos, el porcentaje de acierto es del 77.42%, mientras en el problema 1, el porcentaje de acierto era del 100%.



(a) SI X es label2
ENTONCES Y es Clase-A

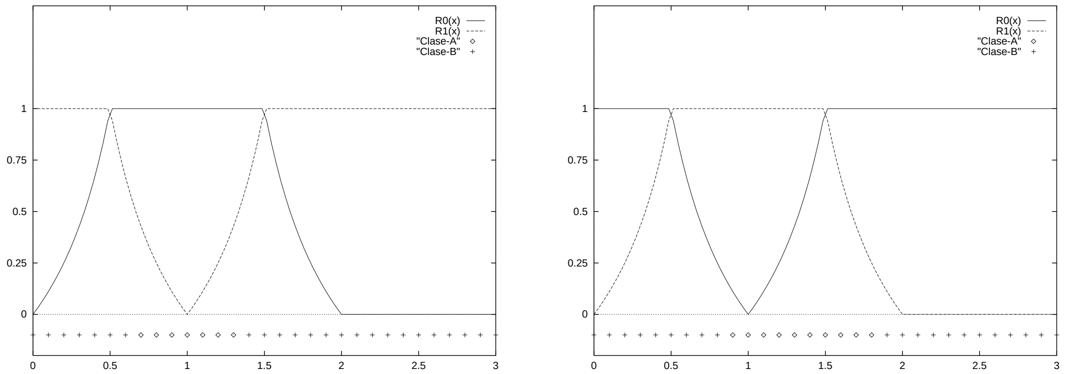


(b) SI X es $\{$label1, label3, label4 $\}$
ENTONCES Y es Clase-B



(c) Interacción entre las reglas para resolver el problema de clasificación

Figura 5.2: Reglas obtenidas por SLAVE



(a) Problema 2 (b) Problema 3

Figura 5.3: Reglas obtenidas por SLAVE para los dos problemas

Los ejemplos anteriores ponen de manifiesto que existe una fuerte dependencia entre la discretización que se realiza sobre los dominios de las variables involucradas y la capacidad de clasificación del algoritmo empleado. Por tanto, si se desea incrementar la capacidad de aprendizaje de los algoritmos que se basan en el MDD, normalmente se utiliza un algoritmo de ajuste final del conocimiento obtenido. Por consiguiente, un algoritmo que sigue esta línea de actuación, parte de una discretización inicial del rango de las variables continuas para obtener un conocimiento inicial. En un proceso posterior, se modifica la discretización inicial para refinar este conocimiento utilizando un algoritmo de ajuste. Estos algoritmos de ajuste (*tunning*) se han usado ampliamente en el campo de las reglas difusas aplicadas a la modelización cualitativa de sistemas. Algunos de estos métodos de ajuste que podemos encontrar en la bibliografía son [69].

En principio, podría parecer que añadir un algoritmo de ajuste a aquellos algoritmos que no alteran la discretización de los dominios es una buena solución para resolver este problema. De hecho, algunos algoritmos basados el modelo descriptivo o aproximativo de Mamdani o que trabajan con el modelo de regla TSK, tales como [22, 23], lo añaden como una componente más del proceso de aprendizaje. Sin embargo, la mayoría de estos algoritmos de ajuste presentan algunos inconvenientes, tales como:

1.) Estos algoritmos se desarrollan para refinar el conocimiento extraido, que normalmente esta representado en forma de reglas, en los problemas de la modelización cualitativa de sistemas. Normalmente, el modelo de regla empleado en este campo implica que todas las variables de entrada toman uno y sólo un valor de su dominio (modelo de regla de Mamdani [34]). Por tanto, para algoritmos que utilicen otros modelos de regla, el proceso de ajuste requiere una adaptación, que no es fácil en todos los casos.

2.) Los algoritmos de ajuste tienen como objetivo optimizar el conocimiento para incrementar la capacidad de representación del comportamiento del sistema que desean aprender. Por consiguiente, es posible que las discretizaciones resultantes no tengan nada que ver (en cuanto a similaridad) con la discretización inicialmente propuesta. Esto será un problema, cuando uno de los objetivos del algoritmo de aprendizaje sea extraer conocimiento que sea fácilmente interpretable desde el punto de vista humano.

3.) Una mala discretización no sólo implica una limitación en la capacidad de aprendizaje del algoritmo, sino que además puede incluir conocimiento para tratar de compensar la falta de discriminación de las discretizaciones propuestas. Si trabajamos con un algoritmo de aprendizaje que representa su conocimiento con reglas, este problema conlleva que:

   i) variables que son irrelevantes sean consideradas como relevantes en la descripción de las reglas, y que

   ii) se incluya un conjunto de reglas donde sólo sería necesaria una única regla, si la discretización fuera correcta.

   Normalmente, los algoritmos de ajuste asumen que la estructura del conocimiento es válida, y tratan de buscar la mejor discretización, que, manteniendo la estructura de este conocimiento, mejore su comportamiento y se adapte todo lo posible al sistema que se desea aprender. Con esta forma de actuación, el conocimiento resultante está expuesto a la redundancia y/o a la limitación en la capacidad de predicción y/o al incremento del espacio de búsqueda.

4.) En general, son algoritmos con alto costo computacional.

Por las razones expuestas anteriormente, descartamos la definición de un algoritmo de ajuste, que asociado a SLAVE, permita mejorar su capacidad de predicción.

Otra posibilidad para aumentar la capacidad de representación del conocimiento consiste en la definición de un modelo difuso que combine las ventajas tanto de los MDDs como de los MDAs. En este sentido, un modelo difuso hibrido (MDH) parte de una discretización inicial de las variables propuestas por un experto, al igual que los MDDs, pero que durante el proceso de aprendizaje sea capaz de modificar la semántica de las etiquetas propuestas inicialmente. En cuanto a esta segunda capacidad, el sistema tiene un comportamiento parecido a los MDAs. Sin embargo, a diferencia de éste, los valores que finalmente toman las variables están conectados con los que el experto propuso y maneja habitualmente para reconocer el comportamiento del sistema, Es decir, la idea es transformar las etiquetas tan sólo cuando sea preciso, lo mínimo necesario y asociar los cambios a las etiquetas iniciales.

Por consiguiente, con un MDH conseguimos que el conocimiento obtenido sea fácilmente interpretable, ya que se define en base a pequeñas alteraciones de las etiquetas lingüísticas que propone inicialmente el experto, y permite aumentar la capacidad de representación del comportamiento del sistema gracias a la flexibilidad que se le otorga al algoritmo de poder modificar la semántica de las etiquetas.

En este capítulo, nos planteamos la posibilidad de convertir SLAVE en un MDH. Para lo cuál, estudiamos enriquecer el lenguaje para aumentar la capacidad de representación del conocimiento, conservando las propiedades expuestas en los capítulos anteriores.

El enriquecimiento del lenguaje pasa por la inclusión en el modelo de regla de **modificadores semánticos** [172]. Los modificadores semánticos permiten pequeñas alteraciones en la semántica de las etiquetas definidas inicialmente, es decir, permiten la definición de nuevas etiquetas durante el proceso de aprendizaje, matizando el significado de las etiquetas originales.

En la siguiente sección proponemos un nuevo modelo de regla, con la inclusión de modificadores semánticos.

5.2 Modificadores semánticos

Cuando un experto en un campo determinado realiza la discretización de una variable que se mueve sobre un referencial continuo, puede conocer de forma aproximada el significado que para él tiene una determinada etiqueta lingüística, pero difícilmente la conocerá de forma exacta. Por ejemplo, en la figura 5.4(a) aparece una posible representación de la etiqueta *aproximadamente 3*. Esta etiqueta puede responder al significado que dicho valor lingüístico tiene para un experto y para una variable concreta. Sin embargo, si a dicho experto le proponemos para el mismo concepto lingüístico alguna de las representaciones (b), (c) o (d) que aparecen en las figuras 5.4, seguramente, algunas de ellas podrían parecerle tan válidas como la que él propone. Aún más, es posible que para otro experto la semántica de dicho concepto quede reflejada mejor con alguna de las alternativas sugeridas al primer experto.

En la literatura sobre la modelización de etiquetas lingüísticas podemos encontrar resultados y discusiones sobre la forma más apropiada de representar un concepto lingüístico. Este hecho ponen de manifiesto que diferentes personas, con un mismo nivel de conocimiento sobre el tema, tiene una percepción diferente para una misma etiqueta lingüística.

Sin embargo, si se toman dos expertos que han realizado diferentes representaciones de un determinado concepto, y se establece una comunicación entre ellos, donde ese concepto es mencionado directamente, ambos colegas son capaces de comprenderse. Esto indica que la representación mental sobre un concepto lingüístico no es tan importante como la noción aproximada de dicho concepto. En este sentido, todas las representaciones

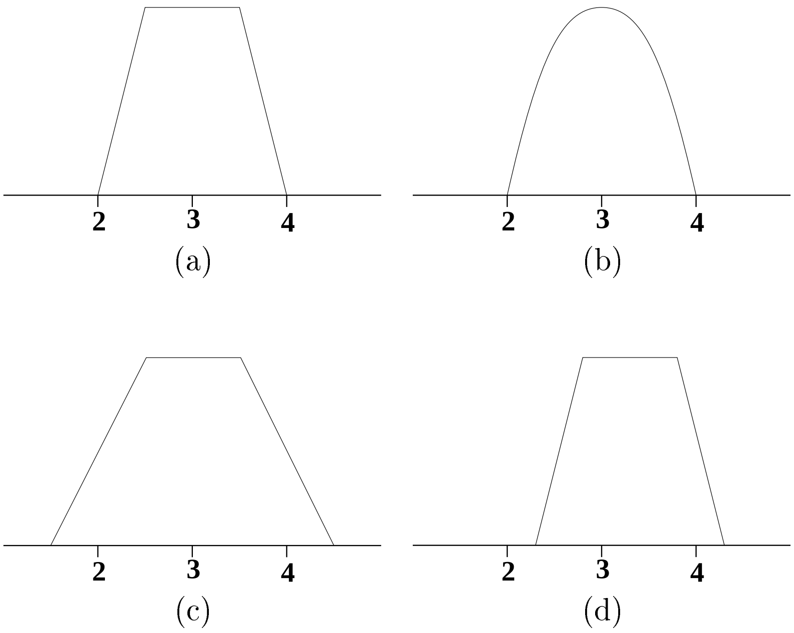


Figura 5.4: Etiqueta lingüística *aproximadamente 3*

que aparecen en la figura 5.4 y muchas más serían igualmente válidas para la etiqueta lingüística *aproximadamente 3* en un proceso de comunicación entre expertos.

Por el contrario, en procesos de extracción automática de conocimiento, la representación semántica de un concepto lingüístico tiene un papel fundamental, ya que para ese concepto se establece una valoración numérica sobre su idoneidad. En este caso, ya no todas las representaciones posibles de un concepto lingüístico, aunque todas representen una noción similar del concepto, son igualmente apropiadas para formar parte del conocimiento que se desea extraer. En este sentido, los MDD encuentran una fuerte restricción para mejorar su capacidad de representar el comportamiento de un determinado sistema.

Por esta razón, parece conveniente dotar a los algoritmos de aprendizaje basados en modelos difusos de la capacidad de alterar la semántica de las etiquetas que componen los dominios de las variables, con el fin de determinar el significado más correcto de la descripción del sistema.

En el campo de la lingüística, los encargados de matizar el significado de un determinado concepto son los modificadores semánticos. Los modificadores semánticos son elementos que aparecen habitualmente en los procesos de comunicación estructurada de alto nivel que permiten ampliar o restringir el significado de una etiqueta lingüística. Estos elementos juegan un papel importante en la comunicación humana, y ya Zadeh en [172] destaca su importancia y su utilidad en la representación del conocimiento en los procesos de razonamiento aproximado.

En modelos de representación del conocimiento basados en reglas difusas, los modificadores semánticos, en muchos casos, aparecen como funciones que permiten alterar la definición de las funciones de pertenencia de las etiquetas difusas. Ya que las etiquetas difusas establecen una distribución de posibilidad de pertenencia al mismo sobre el referencial, los modificadores alteran dicha distribución.

Por tanto, consideraremos un modificador semántico como una función β definida del conjunto de etiquetas al intervalo [0,1], es decir,

$$\begin{array}{cccc} \beta\colon & D & \longrightarrow & [0,1] \\ & \mathrm{L(x)} & \longrightarrow & \beta(L(x)) \end{array}$$

En consecuencia, un MDH parte de un conjunto inicial de etiquetas lingüísticas T también denominados *términos atómicos* que establecen una primera discretización del referencial de las variables. Además, se tiene un conjunto de modificadores semánticos $S = \{\beta_1, \beta_2, \ldots, \beta_m\}$ que permiten alterar la distribución de posibilidad asociada a los términos atómicos, de manera que el modelo de regla asociado es

$$\text{IF } X \text{ IS } \phi(A) \text{ THEN } Y \text{ IS } B$$

donde $\phi(A)$ representa la composición de funciones $\beta_1 \circ \beta_2 \circ \ldots \circ \beta_t$ que son aplicables a las etiquetas de la variable X.

Ejemplo 5.3 *Supongamos que EDAD es una variable lingüística que tiene asociada los siguientes conjuntos de términos atómicos y de modificadores semánticos:*

$$T = \{\mathit{JOVEN,\ ADULTO,\ VIEJO}\}$$

$$S = \{\mathit{MUY,\ MAS\ O\ MENOS,\ ALGO\ MAS\ QUE,\ ALGO\ MENOS\ QUE}\}$$

En base a los conjunto anteriores, es posible construir nuevas etiquetas lingüísticas como:

MUY JOVEN	*MAS O MENOS JOVEN*
ALGO MAS QUE ADULTO	*ALGO MENOS QUE VIEJO*
ALGO MAS QUE MUY VIEJO	*MUY MUY VIEJO*

Del ejemplo anterior, se deduce que la inclusión de los modificadores semánticos amplian el conjunto de valores que puede tomar una variable lingüística y por consiguiente, incrementa la potencia de descripción de los MDH sobre los MDD.

Sin embargo, es posible que algunas combinaciones entre modificadores semánticos y términos atómicos no sean aplicables o interesantes. Por esta razón, asociado a ellos se han de incluir *reglas sintácticas* que establezcan qué modificadores y en que orden pueden ser aplicados a los términos atómicos y *reglas semánticas* que determinen el significado de las diferentes combinaciones.

En función de la complejidad que permitan las reglas sintácticas para la generación de nuevas etiquetas, podemos clasificar los MDH en las siguientes categorías:

Simple: Sólo permite la aplicación de un único modificador semántico para cada término atómico.

MUY JOVEN
ALGO MAS QUE ADULTO
ALGO MENOS QUE VIEJO
MAS O MENOS JOVEN

Simple redundante: Sólo permite la aplicación de un único modificador semántico para cada término atómico, pero este puede ser aplicado un número finito de veces.

MUY MUY JOVEN
ALGO MAS QUE ALGO MAS QUE ADULTO
MUY MUY MUY VIEJO

Complejo con orden fijo: Permite la aplicación de todos los modificadores semánticos sobre cada etiqueta, pero como máximo, éstos sólo pueden aparecer una única vez. Además, el orden en que los modificadores se componen para obtener la nueva etiqueta es fijo.

ALGO MAS QUE MUY JOVEN
ALGO MENOS QUE MAS O MENOS ADULTO
MAS O MENOS MUY VIEJO

Complejo redundante con orden fijo: Permite la aplicación de todos los modificadores semánticos sobre cada etiqueta, y cada uno de ellos puede aparecer un número finito de veces, aunque el orden de composición de los operadores es fijo.

ALGO MAS QUE MUY MUY JOVEN
ALGO MENOS QUE MAS O MENOS MAS O MENOS ADULTO
MAS O MENOS MAS O MENOS MUY VIEJO

Complejo redundante con orden variable: Permite la aplicación de todos los modificadores semánticos sobre cada etiqueta, y cada uno de ellos puede aparecer un número finito de veces, y no existe un orden predefinido para la composición de los modificadores.

MUY MAS O MENOS MUY JOVEN

MAS O MENOS MUY MUY JOVEN

MAS O MENOS ALGO MAS QUE MUY MAS O MENOS MUY VIEJO

Por supuesto, en la clasificación anterior, no se considera la aparición de reglas sintácticas que determinen que ciertos modificadores semánticos no pueden componerse, y por consiguiente, no son aplicables simultáneamente sobre una etiqueta. Por ejemplo, en el caso anterior, los modificadores ALGO MAS QUE y ALGO MENOS QUE, tiene significados opuestos. Tampoco se considera la existencia de reglas que determinen que ciertos modificadores semánticos no son aplicables a ciertas etiquetas.

La clasificación propuesta anteriormente, establece diferentes categorías en función del grado de comprensibilidad de las descripciones resultantes por la incorporación de modificadores semánticos. Por consiguiente, podemos entender que esta clasificación establece diferentes restricciones en la capacidad de construcción de descripciones que permiten la variables lingüísticas booleanas (VLBs) [172]. Por tanto, el concepto de VLB establece el marco más general, y por consiguiente, con mayor grado de libertad para la combinación de modificadores semánticos en las descripciones de las reglas. Debemos destacar que el modelo de VLB no puede incluirse directamente en SLAVE, debido que el modelo de regla definido en este algoritmo no permite algunas de las combinaciones que incluye el modelo VLB.

Esta clasificación determina la capacidad de generación de nuevas etiquetas en base a las inicialmente propuestas. Sin embargo, a medida que la combinación de modificadores es más libre, la comprensión del significado de la misma es más compleja desde un punto de vista humano. En este sentido, reglas sintácticas más restrictivas conllevan la generación de nuevas etiquetas más comprensibles, y en muchos sentidos, estas son más convenientes para su utilización en los MDH.

Por otra parte, las reglas semánticas asociadas a la variables lingüísticas establecen el significado de los términos atómicos y de los modificadores semánticos. Como se dijo anteriormente, en este trabajo los modificadores semánticos son funciones que se aplican sobre los términos atómicos que alteran su distribución de posibilidad sobre el referencial de una variable. Podemos clasificar los modificadores en función de la naturaleza de su regla semántica asociada en:

Modelización fija: Son los modificadores que tienen asociada una función determinada que no admite ser alterada. Por ejemplo, si tratamos de modelizar el modificador MUY, podemos asociarle una regla semántica del tipo:

$$MUY\, L(x) = L(x)^2$$

de manera, que la representación semántica asociada a MUY VIEJO es

$$(\mu_{VIEJO}(x))^2$$

Modelización variable: Son los modificadores que tienen asociada una función determinada que admite ser alterada mediante la inclusión de argumentos. Así, podemos modelizar el operador MUY como:

$$MUY^a L(x) = L(x)^a$$

donde a es un valor fijo definido entre $]0,\infty]$.

Con estos modificadores, la representación semántica de $MUY^{1.5}$ VIEJO es

$$(\mu_{VIEJO}(x))^{1.5}$$

Los modificadores semánticos variables aportan dos ventajas fundamentales: por una parte, permiten trabajar con diferentes significados de un modificador semántico, ya que alteran la semántica del mismo. Por otro lado, pueden permitir eliminar la redundancia de aplicación de los modificadores sobre los términos atómicos, ya que por ejemplo MUY MUY ALTO con la semántica fija descrita anteriormente es $(\mu_{ALTO}(x))^4$ que coincide con MUY^4 ALTO.

Sin embargo, estos modificadores presentan un inconveniente importante, restan comprensibilidad al significado de la etiqueta resultante. Este inconveniente puede paliarse en cierta medida, si la ponderación asociada a la alteración del modificador semántico no cambia bruscamente el significado del mismo. Como se comentó al principio de esta sección, para que el concepto lingüístico que se desea comunicar sea comprendido por el otro interlocutor no es tan importante la representación mental del concepto, como que la noción del mismo sea semejante para ambos interlocutores.

Por consiguiente, en este sentido, los modificadores semánticos con modelización variables incrementan la capacidad de representación desde el punto de vista de la extracción del conocimiento, y no alteran de forma brusca el significado original del modificador (si son definidos correctamente) desde el punto de vista de su comprensibilidad. Además, estos modificadores, proporcionan un número potencialmente infinito de cambios en las etiquetas originales.

5.3 Redefinir SLAVE como un modelo difuso híbrido

En este capítulo proponemos la ampliación de la capacidad de representación del comportamiento de un sistema, redefiniendo SLAVE como un MDH. En concreto, deseamos que SLAVE sea un MDH complejo con orden fijo, ya que proporciona un modelo de regla simple que permite al algoritmo de aprendizaje obtener conocimiento más cercano al comportamiento real del sistema. Además, dentro de este modelo de regla, incluímos los operadores semánticos variables. Los modificadores semánticos incluidos en SLAVE se estudian en el siguiente apartado.

5.3.1 Los modificadores semánticos de SLAVE

SLAVE es un sistema de aprendizaje de propósito general. En este sentido, se hace recomendable establecer un conjunto de modificadores semánticos que no siendo muy numeroso, para no incrementar excesivamente el espacio de búsqueda, permita un ajuste correcto de las etiquetas iniciales. Por consiguiente, para la definición de los mismos, atenderemos a las modificaciones básicas que permitirán mejorar la capacidad de aprendizaje del sistema. Así, si suponemos que la etiqueta inicialmente propuesta por un experto para el concepto *aproximadamente 3* es la representada en la figura 5.4(a), las etiquetas que aparecen en las figuras 5.4(b), 5.4(c) y 5.4(d) pueden obtenerse como alteraciones elementales provocadas por modificadores semánticos. Estas tres alteraciones elementales son el **cambio de la forma de la etiqueta**, el **cambio de amplitud de la etiqueta** y el **desplazamiento de la etiqueta** respectivamente.

Existen muchas más alteraciones elementales que las que se proponen en las figuras anteriores, pero en nuestro estudio solo consideraremos las tres anteriormente mencionadas.

5.3.1.1 Modificación de la forma de una etiqueta

Estos operadores permiten modificar la distribución de posibilidad que refleja la función de pertenencia de los elementos del referencial sobre la etiqueta. De una forma más general, podemos realizar la siguiente caracterización de estos operadores:

Definición 5.1 *Dadas dos etiquetas L_1 y L_2, diremos que L_2 es una etiqueta resultante de una alteración de cambio de forma de la etiqueta L_1, que notaremos por*

$$L_2 = ACD(L_1)$$

si el soporte de L_2 coincide con el soporte de L_1, es decir,

$$Sop(L_1) = Sop(L_2).$$

De entre el conjunto de familias de funciones que satisfacen la anterior definición, estamos interesados en aquellas funciones que alteren la distribución de la etiqueta manteniendo la ordenación relativa en la valoración de posibilidad de los elementos pertenecientes al soporte de la misma,. Por ejemplo, si en la etiqueta original L, dados cualesquiera dos elementos del soporte de la misma a y b, se verifica que

$$\mu_L(a) \geq \mu_L(b)$$

entonces un operador β aplicado a dicha etiqueta debe satisfacer que

$$\beta(\mu_L(a)) \geq \beta(\mu_L(b))$$

Estos operadores de alteración de la distribución de posibilidad de la etiqueta son interesantes desde nuestro punto de vista, porque preservan la ordenación de los elementos del soporte de la etiqueta original con respecto a su posibilidad relativa de pertenencia a la misma, y en consecuencia, la etiqueta resultante de la aplicación del operador presenta una semántica ligeramente diferente a la propuesta inicialmente por el experto.

El operador $\beta(L(x), a) = \mu_L(x)^a$ es un operador que representa a una familia de funciones que verifica la definición del operador ACD y además preserva la propiedad anterior. Un caso particular de este operador es aquel que transforma la etiqueta original mediante $\mu_L(x)^2$ que fué inicialmente propuesto en [172] para representar al modificador semántico **MUY**.

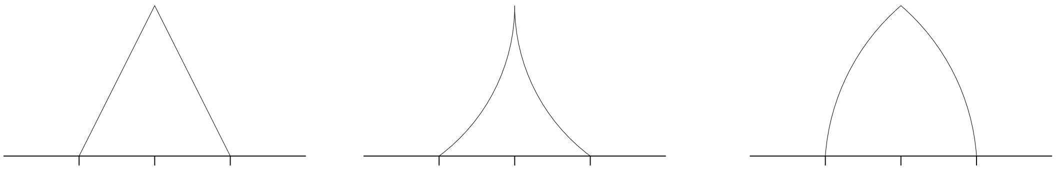
Como se muestra en la figura 5.5, este operador tiene dos interpretaciones diferentes en función del valor del parámetro a. Para valores de $a \in]1, \infty[$, se satisface que $\mu_L(x) \geq \mu_L(x)^a$ para todos los elementos que pertenecen al soporte de la etiqueta. En consecuencia, tras aplicar el operador, aquellos elementos con valores de posibilidad cercanos a uno en la etiqueta original mantienen o decrementan poco su posibilidad de pertenencia a la etiqueta. Por el contrario, los valores de posibilidad cercanos a cero, sufren un decremento más pronunciado en la etiqueta resultante. Por consiguiente, este operador puede entenderse como una modificación en la imprecisión de la etiqueta, ya que los primeros (los que poseen valores cercanos a uno) tras la aplicación del operador, aumentan la posibilidad de pertenecer a la nueva etiqueta en relación a los que poseen valores cercanos a cero en la etiqueta original.

Sin embargo, para valores de $a \in]0, 1[$, se produce el efecto contrario, es decir, la etiqueta resultante aumenta su grado de imprecisión, ya que todos los valores del soporte de la etiqueta mantienen o aumentan su posibilidad de pertenencia a la etiqueta, y este cambio es más evidente sobre aquellos elementos que poseían valores de posibilidad cercanos a cero en la etiqueta original.

Dada un etiqueta cualesquiera L, notaremos este operador de modificación semántica como L^a, donde $a \in]0, \infty[$.

5.3.1.2 Modificación de la anchura de una etiqueta

Este segundo operador elemental de alteración de la etiqueta se puede formular formalmente de la siguiente manera:



Etiqueta Original   Transformación con $a = 2$ Transformación con $a = \frac{1}{2}$

Figura 5.5: Operador $\mu_L(x)^a$

Definición 5.2 *Dadas dos etiquetas L_1 y L_2, diremos que L_2 es una etiqueta resultante de una alteración de cambio en anchura de la etiqueta L_1, que notaremos por $L_2 = ACA(L_1)$, si se verifica que*

$$Sop(L_1) \subseteq Sop(L_2) \qquad o \qquad Sop(L_2) \subseteq Sop(L_1)$$

es decir, el soporte de la etiqueta L_1 está contenido en el soporte de la etiqueta L_2, o es el soporte de la etiqueta L_2 el que está contenido en la etiqueta L_1.

En base a esta definición, la etiqueta resultante al aplicar un operador ACA provoca modificaciones en la distribución de posibilidad de la etiqueta original al igual que en los operadores ACD, pero a diferencia de éstos, el cambio en la distribución de posibilidad viene determinado por la ampliación o reducción del soporte de la etiqueta. Por consiguiente, al igual que el operador anterior, éste también se puede considerar como una modificación que permite aumentar o disminuir la imprecisión de la etiqueta, mediante la inclusión o exclusión de nuevos elementos del referencial en su soporte.

Antes de fijar el operador ACA que utilizará SLAVE, es necesario establecer la siguientes definiciones.

Definición 5.3 *Dada una etiqueta L perteneciente al dominio D de una determinada variable X, que toma valores en un referencial U, cuyos elementos están bien ordenados, definimos*

i) el extremo inferior de la etiqueta, que notaremos por $L^{\triangleleft}$,

$$L^{\triangleleft} = \inf Sop(L)$$

ii) el extremo superior de la etiqueta, que notaremos por $L^{\triangleright}$,

$$L^{\triangleright} = \sup Sop(L)$$

iii) el centro de la etiqueta, que notaremos por $L^{\triangledown}$, como el elemento que se encuentra a igual distancia del extremo superior y del extremos inferior, es decir,

$$L^{\triangledown} = \frac{L^{\triangleright} + L^{\triangleleft}}{2}$$

En base a estos tres puntos que se extraen de una etiqueta, definimos la siguiente familia de funciones que permiten modificar la amplitud de una etiqueta:

Definición 5.4 *Dada una etiqueta L, perteneciente al dominio D de una determinada variable X, que toma valores en un referencial U, cuyo elementos están bien ordenados, y dados los puntos $L^{\triangleleft}$, $L^{\triangleright}$ y $L^{\triangledown}$, el operador ϕ_1^b modifica la etiqueta original de la siguiente forma:*

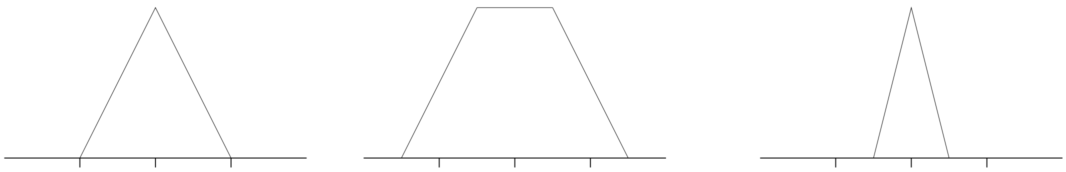
$$\phi_1^b(\mu_L(x)) = \begin{cases} \mu_L(x+b) & si\ x \in [L^{\triangleleft} - b, \min(L^{\triangledown}, L^{\triangledown} - b)] \\ \mu_L(x-b) & si\ x \in [\max(L^{\triangledown}, L^{\triangledown} - b), L^{\triangleright} + b] \\ \mu_L(L^{\triangledown}) & si\ x \in]\min(L^{\triangledown}, L^{\triangledown} - b), \max(L^{\triangledown}, L^{\triangledown} + b)[\\ 0 & en\ otro\ caso \end{cases}$$

En la definición anterior, los valores positivos para el parámetro b provocan una ampliación del soporte de la etiqueta, mientras que los valores negativos para dicho parámetro disminuyen el soporte de la etiqueta original. En esta definición podemos encontrar dos inconvenientes:

a) Mientras que las etiquetas resultantes de la aplicación de este operador, que implican una ampliación del soporte de la etiqueta original, son una distribución de posibilidad, las etiquetas obtenidas tras la aplicación de este operador para reducir el soporte de la etiqueta, no satisfacen en general esta propiedad aún siendo la original una distribución de posibilidad. Por esta razón, modificamos la definición del operador anterior. Por consiguiente, redefinimos el operador de alteración del soporte de la etiqueta, introduciendo una operación de normalización de la siguiente forma:
$$\beta_2^b(\mu_L(x)) = \frac{\phi_1^b(\mu_L(x))}{\max_{u \in U} \phi_1^b(\mu_L(u))}$$

b) Es necesario acotar el máximo grado de modificación si deseamos que las etiquetas resultantes mantengan un significado semejante al de la etiqueta original. En cuanto a la reducción del soporte de la etiqueta, ésta se encuentra acotada, ya que el parámetro b no puede tomar un valor inferior a $L^{\triangleleft} - L^{\triangledown}$. Sin embargo, el proceso de ampliación del soporte, solamente está acotado por el tamaño del referencial. En nuestro modelo deseamos que la etiqueta original pueda reducir su soporte a un único valor, dando la posibilidad al sistema de reducir la imprecisión, pero como máximo que el operador amplie el soporte al doble del de la etiqueta original. Por tanto, establecemos la siguiente redefinición del operador β_2^b como:

$$\beta_2^b(\mu_L(x)) = \frac{\phi^b(\mu_L(x))}{\max_{u \in U} \phi^b(\mu_L(u))}$$



Etiqueta Original   Transformación con $b = \frac{1}{2}$ Transformación con $b = -\frac{1}{2}$

Figura 5.6: Operador L_b

donde ahora

$$\phi^b(\mu_L(x)) = \begin{cases} \mu_L(x + (b \times \delta)) & \text{si } x \in [L^{\triangleleft} - (b \times \delta), \min(L^{\triangledown}, L^{\triangledown} - (b \times \delta))] \\ \mu_L(x - (b \times \delta)) & \text{si } x \in [\max(L^{\triangledown}, L^{\triangledown} - (b \times \delta)), L^{\triangleright} + (b \times \delta)] \\ \mu_L(L^{\triangledown}) & \text{si } x \in]\min(L^{\triangledown}, L^{\triangledown} - (b \times \delta)), \max(L^{\triangledown}, L^{\triangledown} + (b \times \delta))[\\ 0 & \text{en otro caso} \end{cases}$$

siendo $b \in [-1, 1]$ y $\delta = L^{\triangledown} - L^{\triangleleft}$.

Con esta nueva definición, los valores del parámetro $b \in [-1, 0[$ implican una reducción, mientras que para $b \in]0, 1]$ éstos implican una ampliación del soporte de la etiqueta original.

Con las dos matizaciones anteriores, conseguimos restringir las modificaciones permitidas al operador, así como, que cualquier modificación sobre la etiqueta original satisfaga ser una distribución de posibilidad sobre el referencial. En la figura 5.6 mostramos el comportamiento de este operador sobre una etiqueta original de forma triangular.

5.3.1.3 Modificación por desplazamiento de una etiqueta

El tercer tipo de modificación básica que deseamos incorporar a SLAVE es la capacidad de desplazar ligeramente la etiqueta de su ubicación original sobre el referencial. Formalmente, este tipo de operación se puede definir en los siguientes términos:

Definición 5.5 *Dadas dos etiquetas L_1 y L_2, diremos que L_2 es una etiqueta resultante de una alteración de desplazamiento de la etiqueta L_1, que notaremos por $L_2 = AD(L_1)$, si se verifica que*

$$\mu_{L_1}(x) = \mu_{L_2}(x + c) \qquad \forall x \in U$$

donde μ_L representa la función de pertenencia de un elemento x del referencial U a la etiqueta L, y c es un valor real fijo.

En esta definición está incluido cualquier desplazamiento de la etiqueta sobre el referencial, pero al igual que se propuso en el operador anterior, estamos interesados en acotar

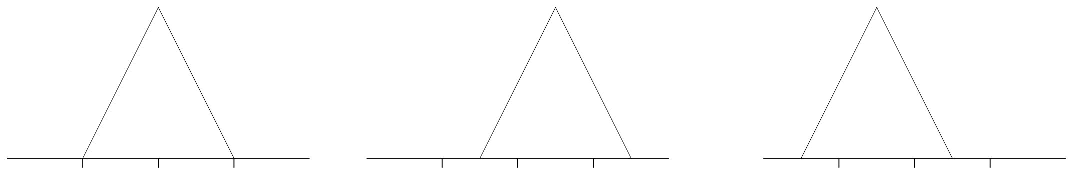


Figura 5.7: Operador $\beta_3(L, c)$

los máximos desplazamientos permitidos. En este sentido, sólo permitiremos desplazamientos de la etiqueta que mantengan en su soporte al menos la mitad de los elementos que pertenencían al soporte de la etiqueta original. Para respetar este criterio, realizamos la siguiente definición de este operador de desplazamiento $\beta_3(L, c)$

$$\beta_3(L, c) = \mu_L(u + c * \delta) \qquad \forall u \in U$$

donde $c \in [-1, 1]$ y $\delta = L^{\triangledown} - L^{\triangleleft}$.

Por tanto, δ representa el máximo desplazamiento permitido de la etiqueta, que coincide con la mitad del soporte de la etiqueta. Los valores positivos para el parámetro c indican desplazamientos hacia la derecha de la etiqueta, mientras que los valores negativos implican desplazamientos hacia la izquierda.

En la siguiente figura, mostramos un ejemplo del comportamiento de este operador sobre una etiqueta original de forma triangular.

5.3.2 Composición de los modificadores semánticos

La inclusión de uno de los anteriores operadores semánticos aumenta considerablemente la potencia de representación del comportamiento de los sistemas. Por ejemplo, si el algoritmo incluyera el operador $\beta_2(L, b)$, podría encontrar una descripción correcta del problema 2 planteado en el ejemplo 5.2 a partir de las etiquetas inicialmente propuestas. Para el valor $b = -0.6$ aplicado a la etiqueta $label2$ en la regla propuesta para la Clase-A, permite una discriminación correcta de las clases involucradas en el problema como muestra la figura 5.8.

Sin embargo, este operador semántico no sería suficiente para discriminar las clases propuestas en el problema 3 del mismo ejemplo. Es más, para este problema, la aplicación de forma aislada de cada uno de los operadores propuestos, no proporcionaría una discriminación correcta.

Este hecho pone de manifiesto que en problemas complejos es necesaria la aplicación de más de uno de los anteriores operadores semánticos para encontrar un correcta descripción

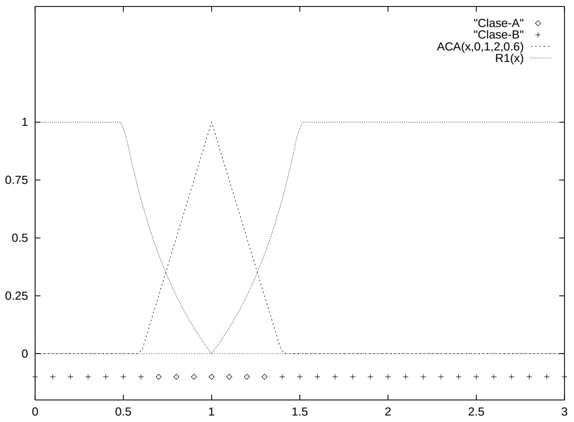


Figura 5.8: Solución al problema 2 usando el operador $\beta_2(label2, -0.6)$

del comportamiento del sistema.

Como ya se ha dicho en este capítulo, cuando se tiene un conjunto de modificadores semánticos aplicables a las etiquetas, se pueden tomar dos opciones: establecer una sintaxis fija en la aplicación de los modificadores sobre las etiquetas, es decir, los modificadores siempre se aplican sobre las etiquetas en un orden previamente fijado, o establecer una sintaxis libre donde esté permitida cualquier combinación en la composición de los modificadores. Antes de tomar partida por una de estas opciones, es necesario estudiar algunas propiedades de los modificadores definidos.

5.3.3 Aproximación genética a la alteración de las etiquetas

La introducción de los operadores semánticos anteriores implica una modificación en la representación del conocimiento extraído, y por consiguiente, una nueva propuesta de modelo de regla. Así, proponemos un modelo de regla que incluye el anterior conjunto de operadores semánticos:

$$\text{IF } X_1 \textit{ is } A_1(a_1, b_1, c_1) \textit{ and } \ldots \textit{ and } X_p \textit{ is } A_p(a_p, b_p, c_p) \text{ THEN } Y \textit{ is } B$$

donde cada variable X_i tiene un conjunto de U_i y toma valores en un dominio finito D_i, para $i \in \{1, \ldots, p\}$. El conjunto de referecia para Y es V y su dominio es F. El valor de la variable Y es B, donde $B \in F$ y el valor de la variable X_i es $A_i(a_i, b_i, c_i)$, donde $A_i \in P(D_i)$ y $P(D_i)$ denota el conjunto de las partes D_i, y a_i, b_i y c_i son los parámetros de los modificadores semánticos β_1, β_2 y β_3 respectivamente. Así,

$$A_i(a_i, b_i, c_i) \equiv \beta_3(\beta_2(\beta_1(A_i, a_i), b_i), c_i)$$

Readaptar SLAVE como un MDH capaz de extraer el conocimiento usando como herramienta el anterior modelo de regla, implica una redefinición del módulo de extracción

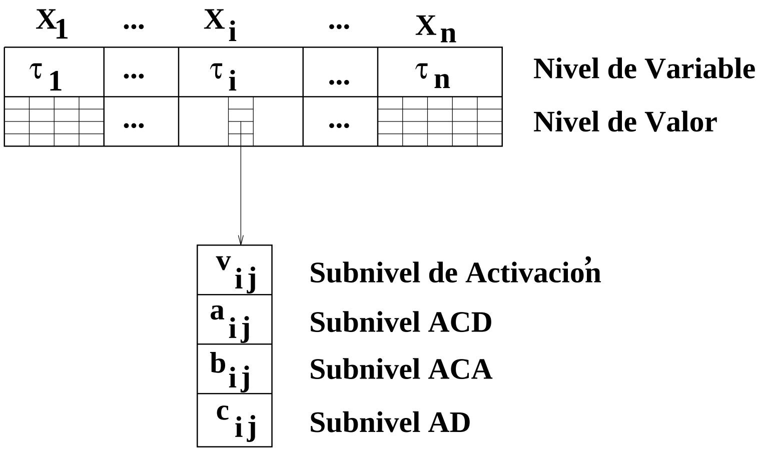


Figura 5.9: Representación del conjunto de soluciones en el AG

de reglas del sistema que viene implementado mediante un algoritmo genético. En esta redefinición, se incorporan como nuevos elementos a buscar, no sólo las variables relevantes para la descripción del concepto y la asignación más apropiada de valores de su dominio, sino que además, forma parte importante del proceso la búsqueda de los valores más apropiados para los parámetros de los modificadores semánticos que se proponen. Por tanto, es necesario incluir en la codificación de los cromosomas la información relativa a los operadores semánticos.

Como se describió en el capitulo anterior, SLAVE usa un algoritmo genético con dos niveles, donde en el nivel de variable se busca el conjunto de variables que son relevantes para describir el concepto, mientras que en el nivel de valor se busca la mejor asignación de valores para las variables relevantes. La mejor solución se busca mediante la coevolución de los dos niveles.

Como los modificadores semánticos afectan directamente al nivel de valor, proponemos la modificación de este nivel, incorporando tres subniveles como muestra la figura 5.9 para buscar la mejor configuración de los modificadores semánticos para las etiquetas asociadas a las variables relevantes.

Con esta representación en dos niveles proponemos nuevamente un modelo basado en la coevolución para encontrar la mejor solución. A diferencia del modelo propuesto en el capítulo anterior, para esta representación aplicamos un modelo basado en la coevolución de cinco algoritmos genéticos, donde la representación de cada nivel o subnivel evoluciona de forma independiente, pero la valoración de una solución se establece en función de la colaboración entre ellos.

Cada subnivel del nivel de valor hace referencia a cada uno de los modificadores semánticos descritos anteriormente. El tamaño de cada subnivel es el mismo y coincide con el número total de etiquetas implicadas en el problema, ya que cada modificador se aplica para todas y cada una de las etiquetas que constituyen los dominios de las variables. Por tanto, todos los subniveles son de longitud fija.

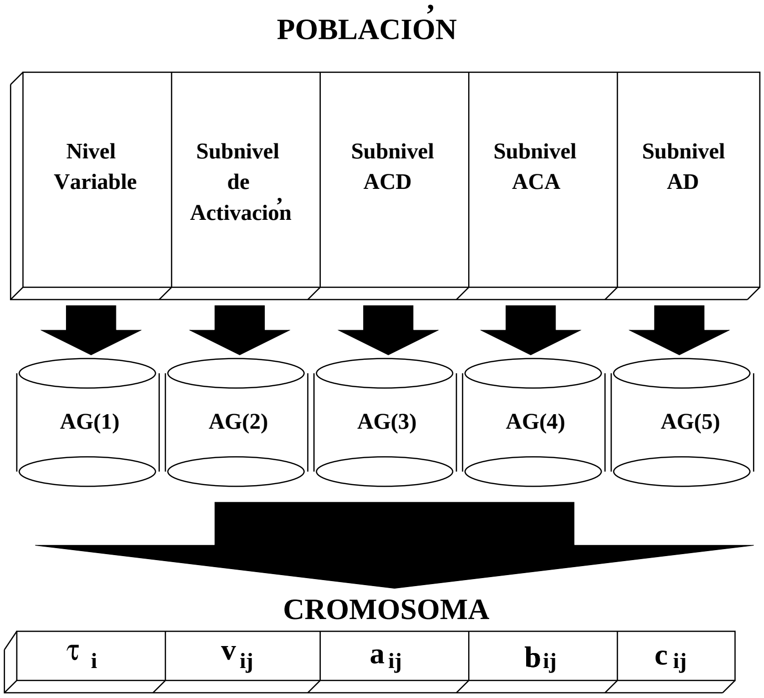


Figura 5.10: Coevolución sobre cinco algoritmos genéticos

El subnivel de activación representa la información relativa a la presencia o ausencia de cada valor en la descripción de la regla. En este sentido, mantiene el significado del anterior nivel de valor, y al igual que esté, conserva la codificación binaria, de manera que el valor uno indica presencia, mientras que el valor cero indica la no consideración de dicha etiqueta.

El subnivel ACD está formado por el trozo de cromosoma que hace referencia a los parámetros del modificador semántico β_1. El rango de valores que puede tomar este parámetro se mueve en el intervalo [0,∞]. Un valor en [0,1[establece un crecimiento de la imprecisión con respecto a la etiqueta original, un valor en]1, ∞], una reducción de la imprecisión, mientras que el valor uno preserva la distribución de la etiqueta original. Es importante destacar que este operador es aplicable únicamenta sobre etiquetas definidas como conjuntos difusos, ya que no altera la semántica de etiquetas definidas como intervalos, es decir, etiquetas donde todos los elementos del soporte de la etiqueta tienen valoración uno.

El subnivel ACA mantiene la información relativa a los parámetros del modificador semántico β_2. Al igual que el modificador β_3, estos parámetros se mueven en el intervalo [-1, 1]. Un valor del parámetro entre [-1,0[o entre]0,1] indican una ampliación o una reducción de la imprecisión con respecto a la etiqueta original respectivamente. No se produce alteración de la etiqueta original si el parámetro toma valor cero.

El subnivel AD codifica los parámetros asociados al modificador β_3 para cada etiqueta. El rango de variación de cada parámetro está en el intervalo [-1,1], donde un valor en

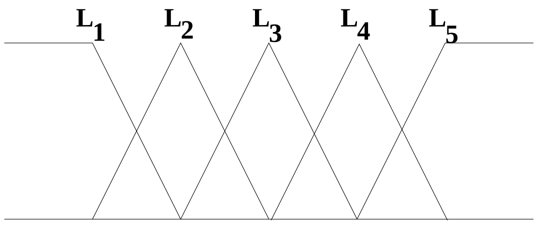


Figura 5.11: Dominio de la variable X

el rango [-1,0[implica un desplazamiento hacia la izquierda con respecto a la etiqueta original, un valor en el rango]0,1] un desplazamiento hacia la derecha, mientras que el cero indica que no se produce ninguna modificación por desplazamiento de la etiqueta original.

Por consiguiente, la semántica de una etiqueta que aparece en la descripción de una regla viene determinada por un tupla de tres componentes (a_i, b_i, c_i), donde a_i, b_i y c_i son los parámetros de los modificadores β_1, β_2 y β_3 respectivamente.

Ejemplo 5.4 *Supongamos que X es una variable considerada relevante para un elemento de la población en la descripción de un concepto. Supongamos además que el dominio asociado a esta variable está compuesto por cinco etiquetas $\{L_1, L_2, L_3, L_4, L_5\}$, siendo la configuración del elemento para esta variable la siguiente:*

	L_1	L_2	L_3	L_4	L_5
v_i	0	1	1	1	0
a_i	1	2	1	1	0.5
b_i	-0.5	0	0	0.25	1
c_i	0.25	0.5	0	0	-0.1

Los valores del subnivel de activación nos indican que las etiquetas que toma como valor la variable X son L_2, L_3 y L_4. Por consiguiente, las otras dos etiquetas no son consideradas y los valores de los parámetros de los modificadores semánticos asociados a ellas no se consideran tampoco. La etiqueta L_3(1,1,0,0) conserva su definición original, ya que los tres parámetros asociados a los modificadores semánticos toman los valores que indican la no alteración de la etiqueta. La etiqueta L_4(1,1,0.25,0) sólamente sufre una alteración provocada por el modificador β_2 que provoca una ampliación del soporte de la etiqueta original. Por el contrario, la etiqueta L_2(1,2,-0.5,0.25) se ve afectada por los tres modificadores semánticos. Así, la configuración de la variable X para este caso concreto es:

$$X \text{ IS } L_1(0,1,-0.5,0.25) \text{ or } L_2(1,2,-0.5,0.25) \text{ or } L_3(1,1,0,0) \text{ or } L_4(1,1,0.25,0) \text{ or } L_5(0,0.5,1,-0.1)$$

(a) Modificaciones aplicadas a las etiquetas (b) Etiqueta resultante normalizada

Figura 5.12: Representación del valor asignado a la variable X

y se puede expresar como:

$$X \text{ IS } L_2(2,\text{-}0.5,0.25) \text{ or } L_3(1,0,0) \text{ or } L_4(1,0.25,0)$$

o de forma más simple como:

$$X \text{ IS } L_2(2,\text{-}0.5,0.25) \text{ or } L_3 \text{ or } L_4(\text{-},0.25,\text{-})$$

De manera que si la figura 5.11 representa el dominio de la variable X, la figura 5.12(a) presenta las modificaciones aplicadas a cada etiqueta activa, mientras que la figura 5.12(b) representa el valor normalizado que toma dicha variable para esta configuración de los parámetros del nivel de valor.

En cuanto a la generación de la población inicial del proceso evolutivo, en el nivel de variable y en el subnivel de activación, este proceso se realiza de forma semejante al descrito en el capítulo anterior, ya que estos dos niveles se heredan directamente de la anterior representación. Con respecto al resto de los subniveles, tomamos el criterio de no alterar inicialmente la definición de las etiquetas de los dominios. Por esta razón, todos los parámetros a_{ij} del modificador semántico β_1 se inicializan a 1, mientras que los parámetros b_{ij} y c_{ij} correspondientes a los modificadores semánticos β_2 y β_3 se inicializan a 0.

Este criterio de inicialización de los parámetros de los modificadores semánticos es justificable, ya que deseamos que las etiquetas utilizadas para representar el conocimiento extraido sean lo más parecidas posibles a las que se proponen inicialmente.

Al igual que sucede con la generación de la población inicial, los operadores genéticos aplicados al nivel de variable y al subnivel de activación coinciden con los ya propuestos en el capítulo. Sin embargo, es necesario establecer los operadores genéticos para los otros tres subniveles, para los cuales únicamente se definen los dos operadores estandar, es decir, el operador de mutación y el operador de cruce.

El criterio utilizado para la construcción de los operadores de mutación se basa en provocar pequeñas modificaciones que consisten en incrementos o decrementos de los parámetros de los operadores semánticos, de manera que el algoritmo genético determine el sentido más apropiado en el que los parámetros deben evolucionar. El uso estricto del criterio anterior provoca que los parámetros de los modificadores semánticos tiendan a tomar valores que modifican las etiquetas originales. Por esta razón, en el operador de mutación aplicado a estos subniveles aumentamos la probabilidad de que cada parámetro tome el valor que no modifica la etiqueta original. El esquema 5.1 representa de forma general el criterio anterior del operador de mutación para el subnivel ACA y AD.

Esquema 5.1 *Esquema general del operador de mutación para el subnivel ACA y AD.*

1. Sea d_{ij} el parámetro de un determinado modificador semántico para la etiqueta j-ésima de la variable i-ésima.
2. Sea p un número generado de forma aleatoria en el rango [0,1]
3. Si $p \in [0, PN[$ entonces $d_{ij} = NE$, donde PN representa un valor entre [0,1[y NE es el valor que permite que el modificador no altere la semántica de la etiqueta inicial.
4. Si $p \in [PN, PN + \frac{PN}{3}[$ entonces $d_{ij} = d_{ij} + \delta$, donde δ representa la variación mínima que se desea provocar sobre dicho modificador.
5. Si $p \in [PN+\frac{PN}{3}, PN+\frac{2\times PN}{3}[$ entonces $d_{ij} = d_{ij}-\delta$
6. Si $p \in [PN + \frac{2\times PN}{3}$,1] entonces d_{ij} no modifica su valor.

El operador de mutación para el subnivel ACD mantiene una estructura semejante, aunque algo más compleja debido al comportamiento que presenta el operador ACD sobre su rango de definición. Así, si el valor del parámetro está en el rango [0, 1[parece lógico que la mínima modificación definida sobre él, sea más pequeña que si el valor del parámetro se encuentra en el rango]1,∞[. Por esta razón, se definen dos valoraciones diferentes de variación mínima sobre el operador, δ_1 asociada a valores en [0,1] y δ_2 para valores en]1,∞[. En este sentido, el esquema anterior es aplicable sin más que modificar el comportamiento de los pasos 4 y 5, de manera que, en función del intervalo en el cuál se encuentre el parámetro a mutar, se sustituirá δ por δ_1 o δ_2.

El operador de cruce definido para estos tres subniveles es el cruce por intercambio sobre dos puntos y coincide con el definido para el subnivel de activación. A pesar de que estos tres subniveles mantienen codificación real, no aplicamos cruces del tipo BLX_α como se hace en el nivel de variable, porque no deseamos que se produzcan fuertes modificaciones

	Codificación	Op. genéticos
Nivel de Variable	Real [0,1]	Cruce BLX_α Mutación no uniforme
Subnivel de Activación	Binaria	Cruce sobre 2 puntos Mutación estandar Operador de rotación Operador de generalización Operador OR Operador AND
Subnivel ACD	Real [0,∞[	Cruce sobre 2 puntos Mutación tipo esquema 5.1
Subnivel ACA	Real [-1,1]	Cruce sobre 2 puntos Mutación tipo esquema 5.1
Subnivel AD	Real [-1,1]	Cruce sobre 2 puntos Mutación tipo esquema 5.1

Tabla 5.1: Características de los algoritmos genéticos implicados en la coevolución

en los párametros de los operadores semánticos, ya que esto aumentaría el tiempo de convergencia del proceso evolutivo, aún a pesar de que posiblemente las descripciones que se obtengan sean mejores.

La tabla 5.1 muestra de forma resumida las características de los algoritmos genéticos implicados en cada nivel del proceso de coevolución.

La característica fundamental del proceso de coevolución que se propone consiste en que no se establece una evaluación individualizada por niveles, sino que existe una única función de evaluación que establece la bondad para cada individuo dependiendo de la constitución global (es decir, de la agrupación de todos los niveles). Debido a esto, el proceso de selección no se realiza de forma individualizada, sino que en función de la valoración de cada individuo se establece un orden lineal sobre todos los elementos de la población. De esta forma, la nueva población se construye seleccionando individuos completos de la última población. Por consiguiente, mientras la aplicación de los operadores genéticos es particular para cada nivel, las operaciones de selección y evaluación son aplicadas a individuos completos.

La función de evaluación usada en SLAVE se definía en base a una EFL sobre varios criterios, que sigue siendo válida para esta nueva redefinición del módulo de selección de reglas, sin más que incorporar en el cálculo, tanto en el grado de satisfacción de la condición de consistencia débil, como en el grado de completitud, las modificaciones provocadas por los modificadores en las etiquetas originales. Además, deseamos incorporar nuevos criterios a la EFL para encontrar la mejor asignación a cada una de las variables relevantes de la descripción de la regla, de manera, que dicha asignación tenga el mayor parecido posible con las etiquetas que se proponían inicialmente.

Se puede establecer un criterio apropiado atendiendo a una medida de distancia, tomando como referencia el valor del operador semántico que no modifica la etiqueta original.

Definición 5.6 *Sean $(d_1, \dots d_n)$ los parámetros de un determinado operador semántico M que afectan a las etiquetas iniciales que aparecen para cada variable en la descripción de un individuo I, y sea b el valor que hace que dicho modificador no afecte al cambio de semántica de la etiqueta. Definimos la diferencia entre las etiquetas resultantes con respecto a las inicialmente propuestas, que notaremos por $Difer_M(I)$, como la distancia entre los parámetros $(d_1, \dots d_n)$ con respecto a $(b, \dots, b)$, es decir,*

$$Difer_M(I) = \sqrt[2]{\sum_{i=1}^{n}(d_i - b)^2}$$

Para aclarar la anterior definición, proponemos el siguiente ejemplo:

Ejemplo 5.5 *Sea I un cromosoma compuesto por dos variables X_1 y X_2 donde cada variable tiene un dominio asociado con tres etiquetas $\{L_{11}, L_{12}, L_{13}\}$ y $\{L_{21}, L_{22}, L_{23}\}$ respectivamente. Supongamos que dicho cromosoma tiene la siguiente configuración:*

Nivel de Variable					
	τ_1	τ_2	τ_{umbral}		
	0.9	*0.1*	*0.6*		
Subnivel de Activación					
L_{11}	L_{12}	L_{13}	L_{21}	L_{22}	L_{23}
0	*1*	*1*	*1*	*0*	*0*
Subnivel ACD					
L_{11}	L_{12}	L_{13}	L_{21}	L_{22}	L_{23}
5.5	*0.5*	*2*	*1*	*0.5*	*3.5*
Subnivel ACA					
L_{11}	L_{12}	L_{13}	L_{21}	L_{22}	L_{23}
-1	*0*	*0.4*	*-0.4*	*1*	*0*
Subnivel AD					
L_{11}	L_{12}	L_{13}	L_{21}	L_{22}	L_{23}
0	*-0.2*	*0.6*	*-1*	*1*	*0.2*

El umbral de activación del nivel de variable τ_{umbral} toma el valor 0.6, por lo que la variable X_2 no aparece en la descripción de la regla y por lo tanto, ninguna de sus etiquetas será computada para determinar la medida de distancia. Por el contrario, la variable X_1 aparece como parte de la descripción, y sus etiquetas activas serán las que utilizaremos para establecer la diferencia. Si nos fijamos en el nivel de activación para *la primera variable, podemos observar que únicamente aparecen en la descripción de la regla las etiquetas L_{12} y L_{13}, y será sobre estas, sobre las que se establezca el cálculo de la distancia.*

Así sobre el subnivel ACA, tenemos que

$$Difer_{ACA}(I) = \sqrt[2]{(0-0)^2 + (0.4-0)^2} = 0.4$$

ya que el cero representa la no modificación de la etiqueta original para este operador. De igual forma podemos calcular:

$$Difer_{AD}(I) = \sqrt[2]{(-0.2-0)^2 + (0.6-0)^2} = 0.632$$

Para el subnivel ACD, es el valor uno el que representa la no modificación de la etiqueta original, por tanto

$$Difer_{ACD}(I) = \sqrt[2]{(0.5-1)^2 + (2-1)^2} = 1.118$$

La medida anteriormente definida establece un criterio claro para valorar la brusquedad de la alteración de las etiquetas con respecto a las que inicialmente se propusieron. En este sentido, incluimos $Difer_{ACD}$, $Difer_{ACA}$ y $Difer_{AD}$ como criterios que deben ser evaluados para establecer la bondad de una descripción, y por tanto, incluidos en la EFL. El problema se plantea al establecer el orden en que estos criterios se deben evaluar. Parece claro, que los criterios ya incluidos en la EFL son más relevantes que estos, y por este motivo mantenemos su orden original. La ordenación de estos tres nuevos criterios entre sí puede ser discutible, pero nosotros optamos por la siguiente: en primer lugar minimizaremos las alteraciones del operador AD, que desde nuestro punto de vista, es el que provoca modificaciones más bruscas sobre la semántica de las etiquetas originales. En segundo lugar, la minimización de las modificaciones del operador ACA y en último lugar las del operador ACD. Situamos en último lugar el operador ACD, ya que es el único modificador semántico que no altera el soporte de las etiquetas iniciales, y por tanto, no provoca cambios bruscos en el significado de las etiquetas.

Por consiguiente, la nueva función $Fitness$ de evaluación de individuos queda definida como una EFL con seis criterios ordenados de la siguiente forma: Dada una regla $R_B(d)$, definimos la valoración de la regla $R_B(d)$ como una EFL con los siguientes criterios:

Orden	Criterio
1	$\max\{\Lambda(R_B(d)) \times \Gamma_{k_1k_2}(R_B(d))\}$
2	$\max\{S(R_B(d))\}$ o $\max\{1 - S(R_B(d))\}$ (discriminante / descriptiva)
3	$\max\{CP(R_B(d)\}$
4	$\min\{Difer_{AD}(d)\}$
5	$\min\{Difer_{ACA}(d)\}$
6	$\min\{Difer_{ACD}(d)\}$

Parámetro	Valor
Número máximo de iteraciones	1000
Número máximo de iteraciones sin mejora	200
Número de individuos en la población	20
Probabilidad de mutación (nivel de valor)	0.001
Probabilidad de cruce (nivel de valor)	0.1
Parámetro de consistencia k_1	0
Parámetro de consistencia k_2	1
Parámetro de recubrimiento λ	0.8
Probabilidad del operador de rotación	0.1
Probabilidad del operador de generalización	0.1
Parámetro de penalización de errores definitivos	1.0
Probabilidad de mutación (nivel de variable)	0.01
Parámetro α del operador BLX_α	0.05
Probabilidad de mutación (nivel ACD, ACA y AD)	0.05
Probabilidad de cruce (nivel ACD, ACA y AD)	0.1

Tabla 5.2: Parámetros fijados para la ejecución de SLAVE con modificadores semánticos

5.4 Pruebas experimentales

Una vez planteado el modelo de aprendizaje con la inclusión de los modificadores, y definidos tanto los operadores genéticos para cada subnivel, como los nuevos criterios en la evaluación de las reglas, deseamos comprobar experimentalmente el comportamiento de este algoritmo de aprendizaje.

Antes de definir el planteamiento general de la experimentación, es necesario realizar algunas consideraciones previas sobre el algoritmo. El hecho de incluir en el proceso las búsqueda de los valores más adecuados de los tres modificadores semánticos que se han definido, provoca un aumento considerable del espacio de búsqueda de las soluciones deseables. Por consiguiente, el número de iteraciones requeridas para que el algoritmo genético converja hacia buenas soluciones debe ser mayor que el utilizado para las anteriores pruebas experimentales. Por esta razón, se han modificado los parámetros propuestos para esta experimentación. Dichos parámetros se muestran en la tabla 5.2.

Los resultados obtenidos usando el algoritmo SLAVE dos niveles con umbral de activación y los anteriores operadores sobre las bases de datos con consecuente crisp se muestran en la tabla 5.3. No se incluyen las bases de datos MONK1, MONK2 y MONK3, ya que todas sus variables predictivas son nominales, por lo que los modificadores semánticos no afectan. Los resultados obtenidos sobre las bases de datos con consecuente difuso se muestran en la tabla 5.4.

Los resultados expuestos muestran claramente una tendencia del algoritmo a ajustarse mucho a los ejemplos del conjunto de entrenamiento. Como se comento en el algoritmo de

Bases de datos	Entrenamiento	Prueba	N.Reglas	Tiempo
IRIS	98.74	92.09	4.4	167
INFARTO	96.70	89.47	2.8	65
WINE	99.34	91.64	4.8	195
PIMA	93.06	68.75	40	9905
IONOSPHERE	99.86	89.38	6	485

Tabla 5.3: Resultados de SLAVE con modificadores semánticos para el caso crisp

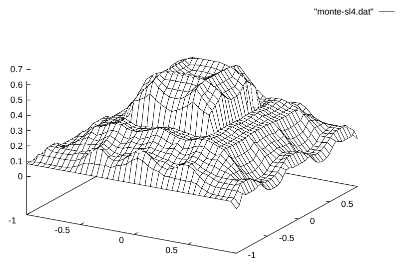
Bases de datos	Entrenamiento	Prueba	N.Reglas	Tiempo
SIN	0.0005	0.0005	7	310
PARABOLA	0.0005	0.0005	3	214
3DPARABOLA	0.0003	0.0009	7	8117
MONTE	0.0009	0.0011	13	18501
PENDULO	0.0978	0.1314	6	512
RICE	0.0038	0.0090	5.6	317

Tabla 5.4: Resultados de SLAVE con modificadores semánticos para el caso difuso

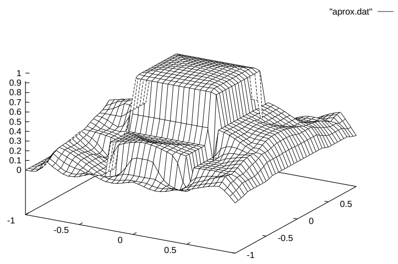
refinamiento, propuesto en el capítulo 2 de esta memoria, este hecho conlleva en la mayoría de los casos a un sobreaprendizaje, provocando una disminución del porcentaje de acierto en los conjuntos de prueba. Justamente esta situación es la que se produce sobre las bases de datos con variable consecuente crisp. Un ejemplo claro de este comportamiento, lo podemos observar sobre la base de datos PIMA, donde se produce un procentaje de acierto del 93.06 % sobre el conjunto de entrenamiento (el más alto de todos los obtenidos sobre este conjunto de ejemplos), mientras que sólo un 68.75 % sobre el conjunto de prueba (el peor de todos los resultados obtenidos sobre esta base de datos por SLAVE).

Con respecto al número de reglas, podemos observar que, exceptuando el elevado número de reglas obtenido para la base de datos PIMA, sus valores coinciden prácticamente con los obtenidos con la versión de SLAVE dos niveles con información, e incluso para algunas bases de datos, este número es inferior al obtenido al aplicar a SLAVE el algoritmo de refinamiento.

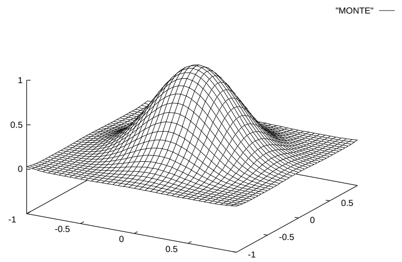
Sobre las bases de datos con variable consecuente difusa, se puede observar una mejora sustancial en la disminución del error sobre el conjunto de entrenamiento, pero al contrario de lo que sucede en el caso crisp, la mejora también se observa en la disminución del error sobre el conjunto de prueba. Esta aparente diferencia de comportamiento entre el caso crisp y el caso difuso del algoritmo tiene una explicación: en las bases de datos crisp, el parecido entre el conjunto de entrenamiento y el conjunto de prueba es menor que el parecido existente entre ambos conjuntos de las bases de datos difusas. Por tanto, si se produce un sobreaprendizaje del conjunto de entrenamiento, el conocimiento extraido representa mejor a aquellos conjuntos de prueba que se asemejan lo más posible al conjunto



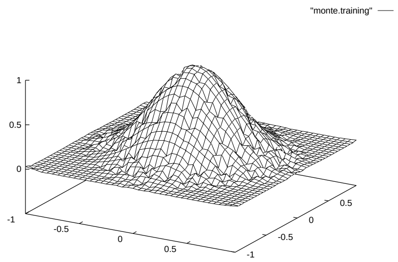
(a) Aproximación obtenida por SLAVE dos niveles con umbral de activación



(b) Aproximación obtenida por el algoritmo WM



(a) Función original



(b) Aproximación obtenida por SLAVE con modificadores semánticos

Figura 5.13: Distintas aproximación obtenidas para la base de datos MONTE

de ejemplos sobre el que se ha aprendido.

Obviamente, al aumentar el espacio de búsqueda, y el número de iteraciones en el algoritmo genético, los resultados del tiempo de respuesta del algoritmo de aprendizaje no son comparables con las versiones propuestas en el capítulo anterior. Sin embargo, si que podemos comparar este parámetro, con los obtenidos por el método de refinamiento propuesto en el capítulo 3, pudiendo observar que, en general, esta nueva aproximación necesita menos tiempo para encontrar el conjunto de reglas que describen el sistema.

De los resultados obtenidos en el caso difuso, debemos destacar la mejora sustancial que se produce con sobre la bases de datos MONTE. En la figura 5.13 se muestran distintas aproximación obtenidas sobre esta base de datos por distintos algoritmos y la función original.

Podemos observar, que el algoritmo SLAVE con los modificadores semánticos propuestos en este capítulo obtiene una aproximación bastante mejor que las que obtienen las versiones anteriores y la que obtiene el algoritmo WM. Además, la similitud entre esta propuesta y la función original, indica que esta aproximación representa correctamente la función de partida.

Las resultados experimentales mostrados en esta sección ponen de manifiesto que la inclusión de operadores semánticos en las descripciones de las reglas es una buena vía para obtener un conocimiento que se aproxime mejor al comportamiento real del sistema. Sin embargo, estos resultados muestran que la implementación aqui utilizada tiene una elevada tendencia al sobre-aprendizaje del conjunto de ejemplos. Por consiguiente, nos proponemos como una futura línea de trabajo la reducción de la tendencia al sobre-aprendizaje del algoritmo propuesto.