

Diseño de Funciones de Similitud para el Razonamiento Basado en Casos usando Programación Genética: estudio con problemas sintéticos

Joan Camps, Josep Maria Garrell, Elisabet Golobardes y David Vernet

Resumen— En este artículo se presentan los resultados obtenidos usando Programación Genética con el fin de obtener una función de similitud ad hoc a un problema para un sistema de Razonamiento Basado en Casos. El sistema final usa la función proporcionada por la Programación Genética para resolver el problema concreto.

Con el fin de constatar si el uso de este proceso mejora el grado de clasificación del sistema en fase de explotación, se parte de un conjunto de casos generados sintéticamente. Se verifica que en el peor de los casos el sistema clasificador llegue a clasificar correctamente con el mismo porcentaje obtenido usando las funciones de similitud clásicas. No obstante, lo habitual es que se mejore considerablemente la eficacia del sistema clasificador.

Palabras clave— programación genética, computación evolutiva, razonamiento basado en casos, aprendizaje artificial, clasificación automática

I. INTRODUCCIÓN

EN el mundo de la clasificación automática, el Razonamiento Basado en Casos (CBR, *Case-Based Reasoning*) [14], [1] es una técnica bien representativa. Mediante una función de similitud que permite obtener una medida objetiva de la proximidad entre dos casos cualesquiera, el CBR actúa buscando en la memoria de casos el caso más cercano al nuevo caso que se quiere clasificar. El punto clave del éxito de este método clasificador consiste en disponer de una buena función de similitud. Existe un buen conjunto de estándares que, en general, funcionan bastante bien, aunque en ciertos problemas sería mucho mejor usar otras funciones de similitud específicas. Esta selección de la función de similitud puede ser llevada a cabo por un experto humano. Habitualmente, puede ser una tarea larga y complicada. El porcentaje de aciertos en la clasificación usando estas funciones depende mucho del tipo de problema que se trata y de la representatividad de la memoria de casos que se tiene.

Grup de Recerca en Sistemes Intel·ligents, Departament d'Informàtica, Enginyeria i Arquitectura La Salle, Universitat Ramon Llull, Passeig Bonanova 8, 08022 - Barcelona. E-mail: {joanc,josepmg,elisabet,dave}@salleURL.edu .

En este trabajo se experimenta sobre la posibilidad de obtener funciones de similitud a medida para cualquier problema dado. Usando una combinación entre la Programación Genética (GP, *Genetic Programming*) [9] y el CBR se pretende conseguir estas funciones de similitud *ad hoc* a cada problema de forma automática. En algunos problemas la mejor función de similitud será la conocida distancia Euclidiana, basada en la métrica de Minkowski, pero en otros se conseguirán nuevas fórmulas para medir esta distancia con mejores resultados de clasificación. Los problemas aquí planteados no son, obviamente, problemas reales aunque en el futuro sí se prevé hacerlo.

El artículo se estructura de la siguiente manera. En primer lugar se presenta los trabajos relacionados, para pasar seguidamente a la explicación detallada de las técnicas básicas del CBR y del GP, así como del sistema usado para conseguir las funciones automáticamente. En el apartado 4 se presentan los problemas que se han usado y los resultados obtenidos, para finalizar con las conclusiones y las posibles líneas de trabajo futuro.

II. TRABAJO RELACIONADO

En este apartado se recoge brevemente algunos trabajos cercanos en lo que concierne al estudio de la diversidad de las funciones de similitud o a algunas hibridaciones realizadas entre el CBR y la Computación Evolutiva.

Toda función de similitud calcula un valor que permite obtener una medida de proximidad o distancia entre dos casos dados. Habitualmente es más fácil trabajar con atributos de dominio continuo, como puede ser un dominio numérico, que trabajar con dominios discretos como son los atributos con valores nominales.

Entre las funciones de similitud basadas en la distancia encontramos: la métrica de Minkowski [3], Mahalanobis [12], Camberra, Chebychev, Quadratic, Correlation, y Chi-cuadrado [11], *Context-Similarity measure* [4], *Contrast Model*,

las funciones basadas en hiperrectángulos [15], [5], *heterogenous distance functions* [17], entre otras. De todas éstas se puede destacar, como una de las más usadas, la de Minkowski [3],

$$Sim(Caso_x, Caso_y, r) = \sqrt[r]{\sum_{i=1}^F w_i \times |x_i - y_i|^r} \quad (1)$$

donde $Caso_x$ y $Caso_y$ son los dos casos de los cuales queremos calcular su grado de similitud; F es el número de atributos que describen el caso; x_i , y_i representan el valor i -ésimo del atributo para el $Caso_x$ y $Caso_y$ respectivamente; w_i es el peso del atributo i -ésimo; y r es el grado aplicado a la fórmula.

De todas las posibilidades que ofrece esta métrica, debe destacarse la distancia Euclidiana (Minkowski de grado 2), que se define como:

$$Sim(Caso_x, Caso_y) = \sqrt{\sum_{i=1}^F |x_i - y_i|^2} \quad (2)$$

La GP tiene un uso extendido en la modelización de sistemas partiendo de un conjunto de muestras que ejemplifican el funcionamiento de este sistema, es la llamada regresión simbólica. La hibridación que se presenta parte de esta capacidad de la GP para encontrar aquellos modelos que pueden funcionar como buenas funciones de similitud. En este sentido, hay algunos trabajos que se acercan a la idea del híbrido usado aquí. Debe destacarse el trabajo, basado en las *automatically defined functions* (ADF) [9], [10], sobre la extracción de los atributos más relevantes usando GP para ser aplicados al CBR [2], [13]. También se pueden encontrar otros trabajos que relacionan el CBR con la computación evolutiva con el fin de extraer las características más relevantes del conjunto de atributos [16], [8].

El trabajo realizado en el presente artículo se basa en la hibridación de la GP con el CBR para buscar directamente la función de similitud específica del problema que se trate. Cabe destacar el trabajo previo realizado dentro del dominio médico de la predicción de cáncer de mama [6].

III. COMBINANDO CBR Y GP

En esta sección se describe el sistema híbrido que se ha utilizado en este trabajo. Este sistema se basa en el uso del CBR y la GP. Inicialmente se explica de forma breve estas dos técnicas usadas y posteriormente se detalla cómo funciona el híbrido.

A. El razonamiento basado en casos

El razonamiento basado en casos (*Case-Based Reasoning*, CBR) es una técnica enmarcada dentro del aprendizaje analógico. Inspirándose en la forma en que los humanos resolvemos un problema partiendo del recuerdo que se tenga de experiencias parecidas [14], este método trata de solucionar un nuevo problema a partir de la selección del caso más próximo de todos los que *recuerda*. El método se caracteriza por cuatro fases [1]: la fase de *recuperación* (*Retrieval phase*) de casos anteriores ya resueltos que sean similares al nuevo caso que se desea resolver; la fase de *adaptación* (*Reuse phase*) de los casos recuperados al nuevo caso; la fase de *revisión* (*Revise phase*) de la solución propuesta; y, finalmente, la fase de *almacenaje* (*Retain phase*) de la información relevante obtenida.

En este trabajo nos hemos centrado en el uso de esta técnica para resolver problemas de *clasificación*. El objetivo es encontrar aquel caso más parecido en la memoria de casos y asociar el nuevo caso a la clase del caso escogido. En concreto, de las 4 fases posibles del CBR sólo nos centraremos en la primera, la fase de recuperación, con el fin de encontrar los casos parecidos. No hace falta aplicar ningún tipo de adaptación porque los nuevos casos que se contemplan pertenecen al mismo dominio que los contenidos en la memoria de casos. Tampoco hace falta revisar nada porque no se pretende, durante la búsqueda de nuevas funciones de similitud, almacenar nuevos casos en la base de conocimiento.

Para conseguir buenos resultados en este proceso, es esencial el uso de una buena medida de las distancias entre los casos para poder seleccionar aquél que sea más próximo. Esta medida viene dada por las *funciones de similitud* que se utilicen y es aquí donde se hibrida este sistema con la GP para encontrar funciones de similitud *ad hoc* al dominio del problema que se esté tratando.

En la configuración usada, el sistema CBR realmente no aprende porque la memoria de casos, una vez cargada inicialmente no se modifica más. Por ahora se ha escogido esta configuración con la intención de usar el CBR únicamente como herramienta evaluadora de las posibles funciones de similitud. Aún así, en el futuro también se podría probar el sistema con aprendizaje, o sea, tratando de ampliar la memoria de casos del CBR.

B. La programación genética

La programación genética (*Genetic Programming*, GP) [9] es una técnica de computación evolutiva parecida a los algoritmos genéticos

(*Genetic Algorithms*, GA) [7] pero usando una representación distinta de los individuos de su población.

El funcionamiento de esta técnica se basa en la existencia de una población de individuos (programas) que van evolucionando con el paso del tiempo debido al cruce y a la mutación, básicamente. Usando una medida de *fitness* específica, se obtiene un valor objetivo para cada uno de estos individuos, posibilitando que éstos perduren en la nueva generación o, por el contrario, tiendan a desaparecer de la población. En este caso, los individuos son programas y el *fitness* es una función que permite medir en qué grado el problema que se trata es solucionado.

Las fases que componen el proceso de la GP básicamente son: la *evaluación* de toda la población; la *selección* de los mejores individuos; la *reproducción* de los individuos en la nueva generación y el *cruce* entre algunos de ellos para pasar a formar nuevos individuos para la siguiente generación; y, finalmente, la *mutación* con la que se consigue pequeños cambios en los individuos.

Una de las principales utilidades de la GP es el uso de la regresión simbólica para encontrar la fórmula que modela el funcionamiento de un sistema. En este caso, los individuos de la población se reducen a ecuaciones que modelan el sistema. El cálculo del *fitness* se reduce a valorar en qué medida se aciertan los resultados esperados a partir de las muestras usadas para el entrenamiento del sistema.

Con la hibridación de esta técnica con el CBR se pretende obtener las funciones de similitud usando la regresión simbólica.

C. CBR+GP: descripción

El sistema que se describe a continuación se basa en el uso del CBR conjuntamente con la GP, técnicas descritas en los apartados anteriores. El objetivo principal es conseguir que la GP encuentre una función de similitud suficientemente buena (como mínimo mejor que las funciones de similitud más usadas) para el CBR en un dominio concreto [6].

Como se puede apreciar en la figura 1, desde el punto de vista del CBR, la GP es un sistema que va proponiendo diferentes *funciones de similitud*. Desde el punto de vista de la GP, el CBR es la forma de obtener el *fitness* de cada uno de los individuos.

En lo que concierne a la GP, el algoritmo arranca con una población de funciones generadas aleatoriamente y prosigue con el proceso clásico descrito antes. La característica diferencial es que

en el momento de realizar la evaluación de los individuos se pasa a utilizar el CBR. Para cada función que debe ser evaluada se utiliza el CBR en la configuración que se ha explicado, tratando de clasificar únicamente los casos que se le ha proporcionado (sin almacenarlos). Para cada función se obtiene un porcentaje de aciertos en la clasificación. Este valor se utiliza como el propio *fitness* de la función. Después de tener todas las funciones evaluadas, el proceso evolutivo continúa normalmente con la reproducción y el cruce habituales.

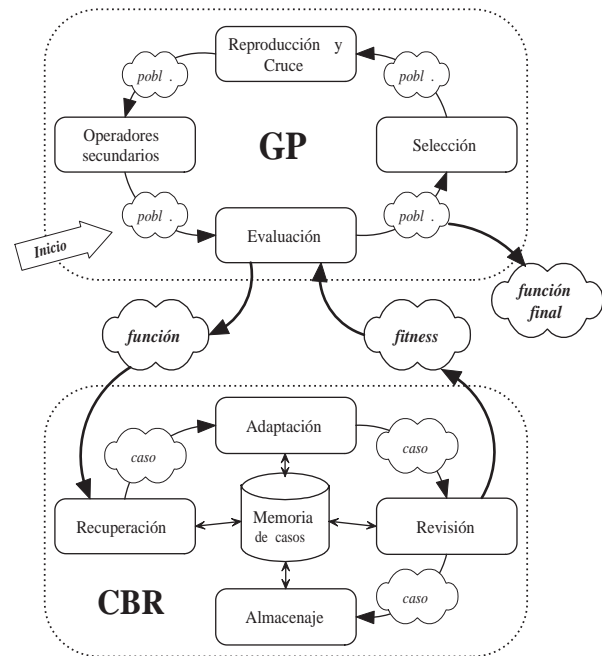


Fig. 1. Sistema híbrido entre CBR y GP.

Al fin de este proceso se obtiene un conjunto de funciones de similitud suficientemente buenas. De todas ellas se escoge la que tiene el mejor *fitness* para usarla como función de similitud del CBR en este dominio.

Al configurar el algoritmo de GP debe definirse una serie de parámetros como son el conjunto de nodos terminales y los nodos no-terminales. Los *terminales* se refieren a los *parámetros* contenidos en las funciones de similitud que van evolucionando. En el híbrido corresponden a los *atributos* del *nuevo caso* y a los *atributos* del *caso recuperado*. A este conjunto se le suele referir como el *conjunto de terminales*. Los nodos no-terminales corresponden a los *operadores* que pueden ser usados en la construcción de las funciones. Es el llamado *conjunto de funciones*. Deberá tenerse en cuenta que el total de los elementos de estos conjuntos sean suficientes como para que se puedan formar las funciones esperadas. Por otro lado, también

es necesario no sobredimensionarlos para limitar mínimamente el espacio de búsqueda de todas las funciones de similitud posibles.

El conjunto de terminales T que se ha escogido incluye los atributos del caso nuevo que se quiere clasificar así como los atributos del caso recuperado de la memoria, el cual se compara con el nuevo. Corresponde al conjunto $T = \{X1, X2, Y1, Y2\}$ (X_i representa los atributos del caso recuperado e Y_i a los atributos del caso nuevo).

El conjunto de operadores O que se ha usado incluye los operadores siguientes: $O = \{+, -, \times, /, \text{absol}, \text{raiz2}, \text{raiz3}, \text{exp2}, \text{exp3}\}$, donde *absol*, *raiz2*, *raiz3*, *exp2* y *exp3* corresponden a los operadores unarios: valor absoluto, raíz cuadrada, raíz cúbica, potencia de grado 2 y potencia de grado 3, respectivamente.

Con estos conjuntos de terminales y funciones se considera que se puede generar las funciones de similitud clásicas que se han descrito antes. No se incluyen las constantes (*ephemeral random constant*) [9], [10] porque se considera que se pueden generar fácilmente con la misma dinámica evolutiva combinando terminales y operadores.

Otros parámetros que se han usado en la configuración seleccionada para el conjunto de pruebas son: la inicialización de la población de forma aleatoria usando el método *ramped-half-and-half* sin decimación, la evaluación de los individuos por el método del *raw fitness, roulette wheel selection* para escoger los individuos que pasan a formar parte de la siguiente generación, y un método de reemplazo generacional con un grado de elitismo de un solo individuo.

Por lo que se refiere al CBR, éste se considera que no aprende porque no almacena nuevos casos en la memoria. De esta forma, la representatividad de los casos que contiene es decisiva para el éxito de la evaluación. En esta línea, la selección que se ha realizado de los elementos de la memoria de casos para cada problema ha sido fruto de una selección precisa con la intencionalidad de conseguir la máxima representatividad de todos los casos posibles del espacio del problema concreto.

IV. EXPERIMENTACIÓN

El objetivo de este trabajo es analizar la posibilidad de obtener funciones de similitud específicas para cualquier problema. Para realizar este estudio se ha usado un conjunto de problemas sintéticos que se describen seguidamente. A posteriori se explicará el juego de pruebas realizadas y los resultados obtenidos.

A. Descripción de los problemas

Se han creado 5 problemas sintéticos de dificultad variada para ver las posibilidades que ofrece el sistema híbrido. Estos problemas nuevos se han generado con la intención de poder asegurar que la memoria de casos en cada uno de ellos se podría considerar completa. Con esto se quiere indicar que seguro que existe una función de similitud perfecta y, además, es conocida a priori para poderla comparar con los resultados obtenidos. En algunos casos el problema se ha diseñado pensando claramente en funciones de similitud estándares tal como la distancia Euclidiana. Otros problemas se han pensado precisamente para que la mejor fórmula no fuera ninguna de las estándares, precisamente para comprobar si era posible encontrar aquel modelo que se había pensado previamente. De la misma forma que la memoria de casos ha sido seleccionada de manera muy precisa, los casos usados para realizar el test del CBR han sido seleccionados aleatoriamente, procurando que todas las clases posibles a clasificar estuvieran representadas. Es importante también resaltar que en ningún momento se ha añadido ruido.

A.1 Problema 1

El problema está definido por 2 atributos (ver figura 2) con valores en el dominio $[-0.3..0.7]$ y $[-0.2..0.8]$. Hay 4 clases posibles. Pensando en una distancia Euclidiana, se ha definido una memoria de 4 casos. Se han usado 50 casos de test. Como se puede observar, el problema es muy regular.

Usando CBR directamente, el porcentaje de aciertos de clasificación de los casos probados es del 100 % tanto para la distancia de Hamming como para la Euclidiana.

A.2 Problema 2

El problema está definido por 2 atributos (ver figura 2), ambos con valores en el dominio $[0..1]$. Hay 7 clases posibles. También, pensando en una distancia Euclidiana, se ha definido una memoria de 18 casos. Se han usado 50 casos de test. El problema deja de ser tan regular como el problema anterior.

El porcentaje de aciertos de clasificación sigue siendo del 100 % con la distancia de Hamming y con la Euclidiana.

A.3 Problema 3

El problema está definido por 2 atributos (ver figura 2) con valores en el dominio $[0..1]$. Hay 5 clases posibles. Siguiendo con la distancia Euclidiana se ha definido una memoria de 16 casos. Se

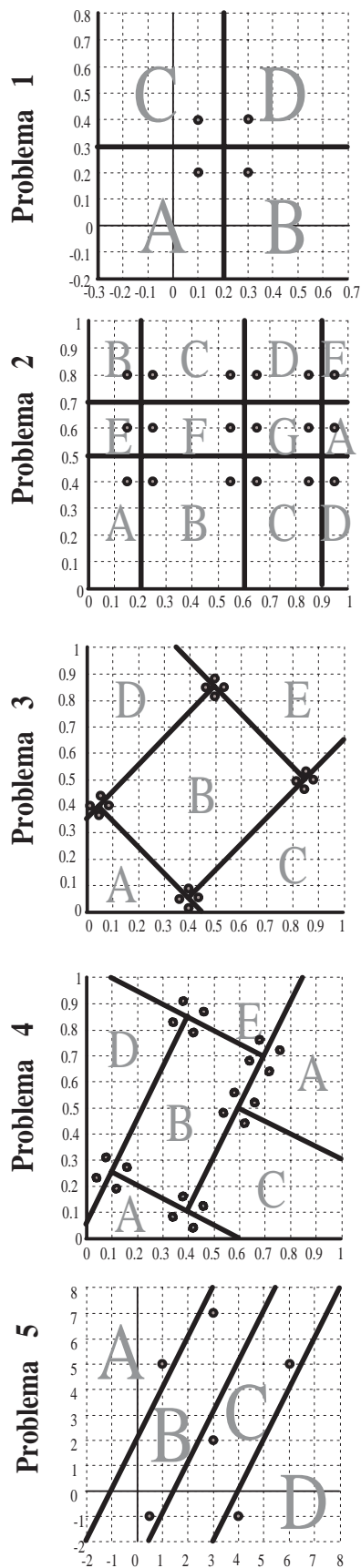


Fig. 2. Representación gráfica de los problemas usados.

han usado 81 casos de test. El problema deja de ser regular y ortogonal.

Con el CBR, el porcentaje de aciertos de clasificación de los casos probados es del 100 % para la distancia Euclidiana pero del 71.6 % para la de Hamming.

A.4 Problema 4

El problema está definido por 2 atributos (ver figura 2) con valores en el dominio $[0..1]$. Hay 5 clases posibles. Se sigue usando la distancia Euclidiana donde se ha definido una memoria de 20 casos. Se han usado 72 casos de test. El problema es poco regular y se han perdido los ángulos de 45° .

Usando CBR directamente, el porcentaje de aciertos de clasificación de los casos probados también es del 100 % para la distancia Euclidiana pero es de un 93.06 % para la de Hamming.

A.5 Problema 5

El problema está definido para 2 atributos con valores en el dominio $[-2..8]$ (ver figura 2). Hay 4 clases posibles. Se prevé que los 6 casos definidos en la memoria de casos no puedan cubrir el 100 % de aciertos usando la función Euclidiana aunque sí se pueden cubrir al 100 % con una fórmula que parte de las proyecciones paralelas a las fronteras del problema. Se trata de ver si el sistema será capaz de encontrar esta u otra función de similitud suficientemente buena usando esa reducida memoria de casos. Se han usado 12 casos de test. Los espacios de las diferentes clases no son equivalentes ni hay ninguna ortogonalidad. Sí se puede observar cierta regularidad en el paralelismo entre las fronteras.

Con el CBR, se ha obtenido un 58.33 % de aciertos con la distancia de Hamming y un 50 % de aciertos con la Euclidiana.

B. Pruebas y resultados

Se han desarrollado dos conjuntos de pruebas para valorar las posibilidades del sistema híbrido con los distintos problemas.

En el primer conjunto de pruebas realizadas, se ha configurado el sistema para trabajar con una población de 400 individuos. Cada ejecución del programa acababa cuando encontraba una función de similitud con un error 0 o cuando llegaba a las 400 iteraciones. Consecuentemente, se considera que no se ha encontrado ninguna función suficientemente buena si se llega a este límite. Se han probado 3 profundidades máximas del árbol que representa la función (4, 5 y 6), con probabilidades de cruce y mutación de 0.25, 0.5 y 0.75. También se ha inicializado el sistema con

TABLA I

PORCENTAJES DE EJECUCIONES CON ÉXITO (QUE HAN OBTENIDO UNA FUNCIÓN DE SIMILITUD CLASIFICANDO PERFECTAMENTE) PARA CADA PROBLEMA Y PRUEBA REALIZADAS.

Problema	Prueba 1	Prueba 2
Problema 1	84.0 %	100 %
Problema 2	58.0 %	83.3 %
Problema 3	11.1 %	20.0 %
Problema 4	13.6 %	10.0 %
Problema 5	51.9 %	76.6 %

3 semillas aleatorias distintas. Todas estas combinaciones de la configuración suponen 81 ejecuciones realizadas para cada problema en este primer conjunto de pruebas.

En el segundo conjunto de pruebas, se ha querido reducir el conjunto de operadores de las funciones con el fin de reducir el espacio de búsqueda posible. Se ha prescindido de las raíces y de los exponenciales. El resto de la configuración ha sido la misma a excepción de la probabilidad de mutación y la de cruce que se han fijado a 0.75 y 0.5 respectivamente. También se ha experimentado con una profundidad máxima de los árboles de 5 y 6. Se han realizado 15 ejecuciones para cada una de las dos configuraciones con lo que se ha podido realizar un conjunto de 30 ejecuciones para cada problema.

Las ejecuciones se han realizado sobre un Pentium III a 700 MHz. Cada una de ellas tenía una duración distinta dependiendo de las iteraciones realizadas y de la profundidad usada en la configuración de los individuos. En promedio, una iteración ha supuesto una duración aproximada entre 10 y 20 segundos.

B.1 Grado de éxito en la obtención de funciones de similitud

El primer aspecto a valorar es el grado de obtención de buenas funciones de similitud de entre todas las ejecuciones realizadas. Se consideran buenas funciones de similitud aquellas que clasifican perfectamente todos los casos que se han proporcionado como test. Cabe destacar que en todas las configuraciones usadas ha habido ejecuciones que han obtenido modelos con una perfecta clasificación de los casos.

En la tabla I se muestra un resumen de los resultados obtenidos con la realización de los dos juegos de pruebas para todos los problemas. Cada valor mostrado corresponde al porcentaje de las ejecuciones que han conseguido modelos con error 0, o sea, capaces de clasificar perfectamente.

TABLA II

PROMEDIO DE ITERACIONES NECESARIAS DE LOS CASOS CON ÉXITO.

Problema	Prueba 1	Prueba 2
Problema 1	13.8	16.8
Problema 2	96.3	75.9
Problema 3	150.1	150.2
Problema 4	251.3	61.0
Problema 5	142.5	71.6

Es preciso destacar que en todos los problemas y en todas las pruebas se han obtenido diferentes modelos que clasificaban perfectamente. Aún así, no todas las ejecuciones han resultado exitosas con el máximo de las 400 iteraciones de que se disponía. Como se puede ver, el porcentaje de buenas ejecuciones varía dependiendo del grado de dificultad del problema. Mientras que los más regulares consiguen porcentajes incluso del 100 %, en los casos más irregulares el porcentaje es muy inferior. En los problemas 3 y 4, por ejemplo, se ha obtenido sólo un 11 % y un 13 % en el primer conjunto de pruebas. En este caso, por los promedios según la configuración, se ve claramente que la profundidad mínima necesaria es de 5 o mejor de 6. También que la probabilidad de cruce ha de ser necesariamente alta. El problema 5, a diferencia de los otros problemas, obtiene mejores resultados con el primer conjunto de pruebas. Evidentemente, con un número superior de iteraciones el grado de ejecuciones exitosas habría sido superior.

En la tabla II se muestra el promedio de iteraciones que ha sido necesario en las ejecuciones exitosas para obtener una función de similitud que clasificara perfectamente. Se puede apreciar cómo las iteraciones necesarias aumentan con la complicación del problema. Consecuentemente, se hace más lento encontrar buenas soluciones con los problemas complicados que con los sencillos.

Por lo que se refiere a la variedad de configuraciones usadas, se ha podido comprobar que hace falta un porcentaje de cruce relativamente elevado. La mutación ha de ser un término medio o elevado para obtener buenos resultados. La profundidad depende mucho del problema en si: si el problema es resoluble con una función muy simple, mejor poca profundidad, pero si se prevé que tendrá cierta dificultad, es mejor darle más profundidad. Normalmente será mejor configurar el sistema con una profundidad grande ya que ante un nuevo problema no se conoce a priori su grado de dificultad en clasificar correctamente. Si

nos referimos a la población, debe observarse que cuanto más profundidad máxima del árbol que representa el individuo, mayor será el espacio de búsqueda y, por tanto, sería necesaria más población para conseguir más variedad genética.

B.2 Las funciones de similitud obtenidas

Debe recordarse que todas las funciones de similitud que se presentan han obtenido un 100 % de porcentaje de clasificación.

■ *Problema 1:* En este problema, el conjunto de funciones obtenidas ha sido muy grande y muy variado pero la gran mayoría de ellas se basaban en el producto del diferencial de los atributos. La función más repetida justamente ha sido ésta:

$$FS(X, Y) = |(X_1 - Y_1)(X_2 - Y_2)| \quad (3)$$

En este caso también nos ha aparecido claramente la distancia de Hamming y la Euclidiana pero sin aplicar la raíz:

$$FS(X, Y) = (X_1 - Y_1)^2 + (X_2 - Y_2)^2 \quad (4)$$

■ *Problema 2:* Se ha vuelto a conseguir la distancia Euclidiana y algunas variaciones entre la distancia Euclidiana y la de Hamming como es:

$$FS(X, Y) = |(X_1 - Y_1)| + (X_2 - Y_2)^2 \quad (5)$$

Ha aparecido una gran variedad de funciones posibles pero todas se basan en las distancias de los atributos.

■ *Problema 3:* Entre la gran variedad de funciones resultantes, se ha obtenido la función de la distancia Cúbica así como la siguiente función:

$$FS(X, Y) = (X_2 - Y_2)^2 - (X_1 - Y_1)^2 \quad (6)$$

■ *Problema 4:* En este problema, ya menos regular que los anteriores, la función que ha aparecido repetidamente es la función Euclidiana como en el caso del problema 2 con la ecuación 4 y funciones muy parecidas a ésta.

■ *Problema 5:* En este problema que, intencionadamente, ya se ha generado con datos que no iban demasiado bien para las funciones estándares, se ha obtenido la siguiente función en un tercio de las ejecuciones exitosas:

$$FS(X, Y) = |2X_1 - 2Y_1 - X_2 + Y_2| \quad (7)$$

Esta función no es otra cosa que la diferencia entre las proyecciones de los puntos sobre el eje de las abscisas siguiendo paralelamente a las fronteras de las diferentes áreas.

V. CONCLUSIONES Y TRABAJO FUTURO

El objetivo inicial del trabajo era diseñar automáticamente funciones de similitud para ser usadas con el CBR en un problema concreto. La intención era encontrar la mejor función de similitud para cada caso. Todos los problemas usados podían ser resueltos con un 100 % de porcentaje de aciertos de clasificación usando la función adecuada. Cabe destacar que en todos los problemas se ha obtenido funciones de similitud con error 0, clasificando perfectamente.

Se ha podido comprobar cómo en los problemas más generales, como son el problema 1 y problema 2, se obtenían funciones genéricas (Hamming, Euclidiana y Cúbica), mientras que para problemas más concretos el sistema era capaz de encontrar una función de similitud *ad hoc* como es el caso del problema 5. En este caso se ve claramente cómo las funciones genéricas no daban buenos resultados y era necesario el diseño de funciones más específicas. Debe tenerse en cuenta que encontrar estas buenas funciones de similitud *manualmente* puede ser una tarea difícil mientras que con la automatización se obtienen fácilmente.

Los resultados conseguidos validan el sistema como una buena herramienta para obtener estas funciones. Se ha visto que dependiendo de la *dificultad* del problema la obtención de buenas funciones resulta más fácil o más difícil. La configuración del sistema se ve también como un tema importante para conseguir una buena función.

Las líneas de trabajo abiertas se orientan al perfeccionamiento de este sistema. Por un lado, se podrían mejorar los resultados obtenidos usando técnicas que permitan el mantenimiento de una mayor variabilidad genética entre generaciones, al estilo de la especiación, con tal de facilitar la aparición de la mejor función. También se prevé como una línea de futuro la utilización del CBR pero en configuración de aprendizaje, adecuando la memoria de casos progresivamente según los nuevos casos que se vayan resolviendo. Finalmente, otra posible línea de trabajo es la realización de estos mismos juegos de pruebas pero usando memorias de casos no ideales, o con ruido, y probar el grado de éxito en la obtención de buenas funciones de similitud para problemas reales.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por el proyecto TIC2002-04036-C05-03. Quisiéramos agradecer a Ingeniería i Arquitectura La Salle, de la Universitat Ramon Llull, por el soporte a nuestro Grupo de Investigación en Sistemas Inteligentes.

REFERENCIAS

- [1] Aamodt, A. and Plaza, E., Case-Based Reasoning: Foundations Issues, Methodological Variations, and System Approaches, *AI Communications*, 7 (1994), 39-59.
- [2] Ahluwalia, M. and Bull, L., Coevolving Functions in Genetic Programming: Classification using K-nearest-neighbour, *Proceedings of the Genetic and Evolutionary Computation Conference*, (1999), 947-952.
- [3] Bachelor, B., *Pattern Recognition: Ideas in Practice*, Plenum Press, 1978.
- [4] Biberman, Y., A Context Similarity Measure, *European Conference on Machine Learning*, (1994), 49-63.
- [5] Domingos, P., Context-sensitive feature selection for lazy learners, *AI Review*, 11 (1997), 227-253.
- [6] Golobardes, E., Nieto, M., Salamó, M., Camps, J., Calzada, G., Martí, J. and Vernet, D., Generació de funcions de similitud mitjançant la Programació Genètica pel Raonament Basat en Casos, *Butlletí de l'ACIA, Quart Congrés Català d'Intel·ligència Artificial*, 25 (2001), 100-107.
- [7] Holland, J., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.
- [8] Kelly, J. D. and Davis, L., Hybridizing the Genetic Algorithms and the K Nearest Neighbors Classification Algorithm, *Proceedings of the Fourth International Conference on Genetic Algorithms*, (1991), 377-383.
- [9] Koza, J.R., *Genetic Programming. On the programming of computers by means of natural selection*, Massachusetts Institute of Technology Press, 1992.
- [10] Koza, J.R., *Genetic Programming II. Automatic Discovery of Reusable Programs*, Massachusetts Institute of Technology Press, 1994.
- [11] Michalski, R. and Stepp, R. and Diday, E., A Recent Advance in Data Analysis: Clustering Objects into Classes Characterized by Conjunctive Concepts, *Progress in Pattern Recognition*, (1981).
- [12] Nadler, M., Pattern recognition engineering, *John Wiley & Sons*, (1993), 299-302.
- [13] Raymer, M.L. and Punch, W.F. and Goodman, E.D. and Kuhn, L.A., Genetic Programming for Improved Data Mining: An Application to the Biochemistry of Protein Interactions, *Genetic Programming 1996: Proceedings of the First Annual Conference*, (1996), 375-380.
- [14] Riesbeck, C.K. and Schank, R.C., *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates, 1989.
- [15] Salzberg, S., A nearest hyperrectangle learning method, *Machine Learning*, 6 (1991), 277-309.
- [16] Siedlecki, W. and Sklansky, J., On Automatic Feature Selection, *Proceedings of the International Journal of Pattern Recognition and Artificial Intelligence*, (1988).
- [17] Wilson, D.R. and Martinez, T.R., Improved Heterogeneous Distance Functions, *Journal of Artificial Intelligence Research (JAIR)*, 1 (1997), 1-34.