CONTROLADOR SECUENCIAL SEGMENTADO PARA LÓGICA BORROSA IMPLEMENTADO EN FPGA

Gerardo Aranguren Aramendía
Dpto. de Electrónica y Telec.
E. T. S. Ing. Ind. y Telec.
Universidad del País Vasco
Alda. Urquijo s/n. 48013 Bilbao
jtpararg@bi.ehu.es

Miguel Rodríguez Gómez
Dpto. de Matem. Aplicada
E. T. S. Ing. Ind. y Telec.
Universidad del País Vasco
Alda. Urquijo s/n. 48013 Bilbao
maprogom@bi.ehu.es

Mariano Barrón Ruiz
Dpto. de Ing. de Sist. y Automát.
E. U. Ing. Técnica Industrial
Universidad del País Vasco
Avda. Otaola, 29, 20600 Eibar
ispbarum@sb.ehu.es

Resumen

Para una rápida evaluación de los sistemas basados en reglas borrosas se han propuesto distintas arquitecturas de procesamiento paralelo basado en Circuitos Integrados de Aplicación Específica (ASIC). En este artículo presentamos una serie de simplificaciones de planteamiento de la lógica borrosa, que nos permiten realizar un ASIC con funcionamiento secuencial. Las cualidades de esta nueva arquitectura comparte algunas de las ventajas de los sistemas de tratamiento hardware, velocidad, y algunas de las ventajas de los sistemas software, flexibilidad.

Palabras Clave: Fuzzy, Pipe-line, FPGA

1 INTRODUCCIÓN

Tradicionalmente, las soluciones que se han adoptado para la ejecución de los algoritmos borrosos han sido implementadas mediante software basado en distintos microprocesadores o microcontroladores. Algunas aplicaciones han sido desarrolladas aprovechando cualidades específicas de determinados microprocesadores, buscando de esa forma mejorar el tiempo de tratamiento de la información.

Los trabajos del equipo de Zimmermann [1] son notables en el empleo del Transputer para control borroso, sirviéndose de la capacidad de disponer de varios Transputer trabajando en paralelo. También podemos destacar Eichfeld y otros [2] con el procesador SAE 81C99, que actua como unidad lógica borrosa esclava de un microprocesador maestro.

Pero, tal vez, los trabajos más relacionados con nuestra línea de investigación son los dedicados a desarrollar un ASIC para funciones de control borroso. De estos el pionero fue el trabajo de Togai [3], que propone la primera arquitectura hardware para procesamiento borroso. A este siguió Watanabe [4], que presenta una arquitectura paralela, que sirve de referencia a cualquier desarrollo posterior. Samoladas y Petrou [5] describen dos nuevas arquitecturas paralelas, realizando las operaciones composicional e inferencial sobre unos mismos recursos. Otros trabajos de interés son: Hung [6], Salvador y Gutiérrez [7] y, en la colección de artículos presentados por Kandel [8], los correspondientes a Ascia [9], Jaramillo-Botero [10], Li [11] y Ruiz [12].

En nuestro grupo de investigación hemos desarrollado un controlador borroso intermedio entre los desarrollos basados en microprocesadores, muy flexibles, pero no suficientemente rápidos para determinadas aplicaciones, y los desarrollos mencionados en ASIC con procesamiento paralelo, menos flexibles, pero muy rápidos. La implementación se ha realizado en un dispositivo programable tipo FPGA.

Para la realización de este circuito hemos tenido en cuenta a Pedrycz [13], que en su libro "Fuzzy Control and Fuzzy System" describe las características que debe tener un controlador borroso. Entre otros aspectos indica los cuatro modos de operación de los controladores, nosotros hemos optado por realizar un controlador tipo *Numerical Input Numerical Output*, denominado por otros autores como SISC (*Singleton Input Singleton Consequent*) con aplicación a sistemas de control MIMO (*Multiple Input Multiple Output*).

2 SIMPLIFICACIONES LOGICAS

Consideremos un sistema de control borroso formado por m entradas numéricas (X1, X2, ..., Xm), una salida numérica Z y n reglas:

R1: IF X1 is A1,1 AND X2 is A1,2 AND ... AND Xm is A1,m THEN Z is B1

R2: IF X1 is A2,1 AND X2 is A2,2 AND ... AND Xm is A2,m THEN Z is B2

...

Rn: IF X1 is An,1 AND X2 is An,2 AND ... AND Xm is An,m THEN Z is Bn (1)

Donde los términos Aj,i son literales borrosos.

Para diseñar el controlador borroso vamos a considerar tres simplificaciones en esta formulación:

2. 1 PRIMERA SIMPLIFICACIÓN

Los términos lingüísticos Aj,i sólo pueden representarse como Π-términos, Λ-términos, S-términos o Z-términos (estos dos últimos también denominados literales saturados) (fig. 1). Los trapecios y triángulos vamos a desdoblarlos en dos literales compuestos formados cada uno de ellos por un literal saturado a los que denominaremos: ASj,i y AZj,i según representen la saturación respecto del límite derecho (S-términos) o del izquierdo (Z-términos).

El nuevo sistema de reglas que obtenemos a partir de esta simplificación es:

R1: IF X1 is AS1,1 AND X1 is AZ1,1 AND X2 is AS1,2 AND X2 is AZ1,2 AND ... AND Xm is AS1,m AND Xm is AZ1,m THEN Z is B1

R2: IF X1 is AS2,1 AND X1 is AZ2,1 AND X2 is AS2,2 AND X2 is AZ2,2 AND ... AND Xm is AS2,m AND Xm is AZ2,m THEN Z is B2

...

Rn: IF X1 is ASn,1 AND X1 is AZn,1 AND X2 is ASn,2 AND X2 is AZn,2 AND ... AND Xm is ASn,m AND Xm is AZn,m THEN Z is Bn (2)

Es conocido el modo gráfico de presentación de estas reglas y su resolución por el algoritmo máximo-mínimo (cfr. fig. 8.4. de [14], o fig. 4 de [4]), por tanto, es evidente que cualquiera de las subpremisas "Xi is Aj,i" es equivalente a la composición mediante el operador mínimo de las subpremisas "Xi is ASj,i" y "Xi is AZj,i" (fig. 2), cualquiera que sea el valor de la variable Xi.

$$Xi \text{ is } Aj,i \Leftrightarrow \min \{Xi \text{ is } ASj,i \, , \, Xi \text{ is } AZj,i\} \quad (3)$$

Inicialmente (1) teníamos n reglas cada una de m subpremisas o cláusulas, esta primera simplificación (2) duplica el número de subpremisas, de forma que se requieren examinar a partir de ahora 2.n.m subpremisas.

El interés de la simplificación realizada estriba en que cada literal, inicialmente definido por cuatro números (P, Q, R, S) queda definido por dos parámetros (Z, Y): Z es el límite entre el grado de activación nulo y distinto de nulo, e Y es el límite entre el grado de activación unidad y el menor de la unidad. Esto nos permitirá realizar la operación composicional mediante una tabla de datos (LUT) implementada en una memoria relativamente pequeña.

2. 2 SEGUNDA SIMPLIFICACIÓN

Una de las formas de enunciar el conjunto de reglas, que definen el funcionamiento de un actuador, es mediante una matriz de tantas dimensiones como variables de entrada tenga el sistema a controlar (cfr. cap. 4. de [13], o cap. 3 de [15]). Si disponemos de m entradas, tendremos una matriz de dimensión m. Cada una de las dimensiones tendrá un cardinal igual al número de literales de cada variable, para la variable de entrada Xi tenemos Ci literales distintos. Luego el número de elementos total de la matriz es:

$$n' = C1 \, . \, C2 \, . \, \ldots \, . \, Cm \quad (4)$$

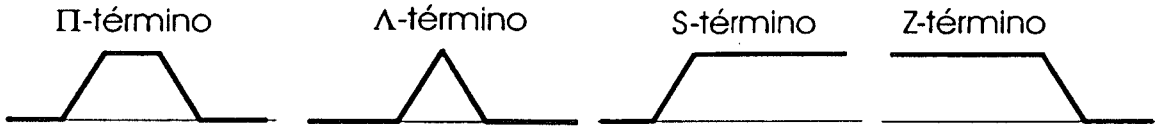

Figura 1. Representación gráfica de términos lingüísticos borrosos.

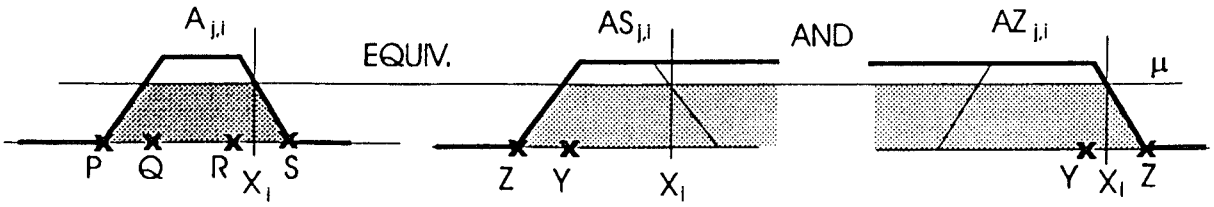

Figura 2. Equivalencia entre un literal borroso y la composición de dos literales saturados.

El planteamiento realizado hasta el momento no contempla la posibilidad de simplificación o la aplicación de la heurística. No obstante, es notorio que algunas de las subpremisas puede que no se consideren en determinadas reglas, es decir, las reglas no tienen porqué estar formadas siempre por m subpremisas, por tanto, podemos reducir el número de subpremisas si se realiza un análisis secuencial.

Del mismo modo, el número de reglas n que definen el sistema (1) pueden no considerar todas las posibles combinaciones de literales, es decir, puede ser:

$$n \leq n' \qquad (5)$$

La consideración de un número de reglas menor de n' reduce el tiempo de ejecución en un sistema secuencial. La definición de un sistema con menos de n' reglas se debe a la heurística aplicada al definir el sistema borroso.

2.3 TERCERA SIMPLIFICACIÓN

En este mismo orden de actuación, podemos añadir un nuevo modo de simplificación, que nos recuerda los diagramas de Karnaugh de la lógica bivaluada.

En un sistema con términos lingüísticos normalizados, consideremos dos reglas de las enunciadas en (2), que tengan igual consecuente y todos los literales antecedentes iguales menos uno que difiere en una unidad, es decir, dos elementos vecinos:

Rj: IF X1 is ASj,1 AND X1 is AZj,1 AND ... AND Xk is ASj,k AND Xk is AZj,k AND ... AND Xm is ASj,m AND Xm is AZj,m THEN Z is Bj

Rj+1: IF X1 is ASj+1,1 AND X1 is AZj+1,1 AND ... AND Xk is ASj+1,k AND Xk is AZj+1,k AND ... AND Xm is ASj+1,m AND Xm is AZj+1,m THEN Z is Bj+1

Donde: ASj,i = ASj+1,i ; AZj,i = AZj+1,i
para i=1, 2, ..., k-1, k+1, ..., m (6)

Si los literales Bj y Bj+1 resultan ser iguales en un determinado sistema, podemos simplificar estas dos reglas. La regla resultante tendrá todos las subpremisas comunes y las dos subpremisas extremas de la posición no común. Dependiendo de los posibles valores de los literales ASj,k, AZj,k, ASj+1,k y AZj+1,k:

a) Si ASj,k < ASj+1,k y AZj,k < AZj+1,k entonces: (7)

Rj&j+1: IF X1 is ASj,1 AND X1 is AZj,1 AND ... AND Xk is ASj,k AND Xk is AZj+1,k AND ... AND Xm is ASj,m AND Xm is AZj,m THEN Z es Bj

b) Si ASj,k > ASj+1,k y AZj,k > AZj+1,k entonces: (8)

Rj&j+1: IF X1 is ASj,1 AND X1 is AZj,1 AND ... AND Xk is ASj+1,k AND Xk is AZj,k AND ... AND Xm is ASj,m AND Xm is AZj,m THEN Z es Bj

Con lo que se reducen en 2.m el número de subpremisas a analizar. En la figura 3 se presenta la simplificación del caso a) en forma gráfica aplicado a dos variables, m=2.

En la figura 4 se muestra como ejemplo, la capacidad de estas simplificaciones para un sistema borroso a partir de un ejemplo extraído de Li [11]. Como se puede apreciar las reglas de simplificación, similares a las de Karnaugh, son:

a) Se pueden agrupar en una sola regla las reglas del mismo consecuente que tengan vecindad, es decir, que se diferencien en un único antecedente y este antecedente tenga unos literales consecutivos en ambas reglas.

b) La generalización de la agrupación de reglas también es válida. Se pueden agrupar en una sola regla las reglas del mismo consecuente que tengan vecindad múltiple, es decir, que los antecedentes en que difieran tengan unos literales consecutivos en ambas reglas.

c) Estas agrupaciones de reglas deben incluir al mayor número posible de reglas, estableciendo grupos con vecindad simple o múltiple que formen un paralelepípedo de cualquier dimensión menor que m y de cualquier longitud de aristas.

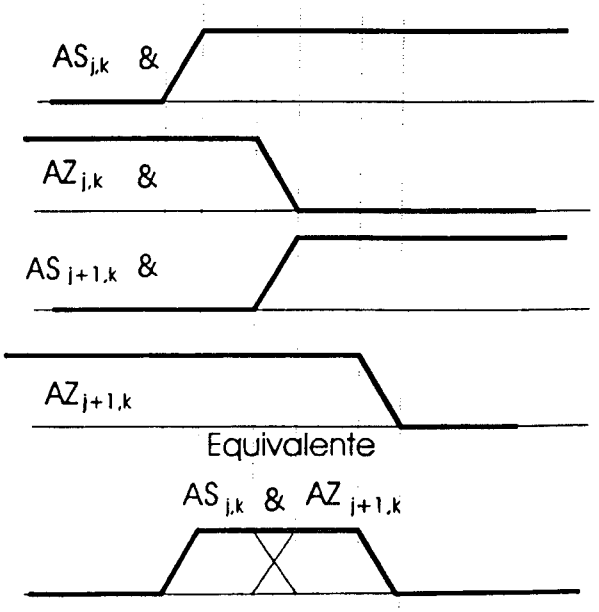

Figura 3. Simplificación de dos subpremisas.

d) Una agrupación de reglas queda definida por sus antecedentes comunes y los literales extremos de los antecedentes no comunes.

e) Todas las reglas iniciales deben estar comprendidas al menos en una de las agrupaciones de reglas.

f) Una regla puede formar parte de varias agrupaciones de reglas siempre que la s-norma cumpla la propiedad idempotente.

g) Los elementos de la matriz no definidos, reglas indiferentes, se pueden asociar a cualquier grupo si favorecen la formación de grupos más grandes de reglas.

h) Los literales de los extremos en la mayoría de los casos son saturados (S-términos o Z-términos), por lo que su inclusión en una nueva regla reduce el número de subpremisas.

i) El número de subpremisas de un grupo es igual a dos veces el número de variables que lo definen. Si un grupo tiene alguna variable saturada se descuenta esta en la definición de las subpremisas.

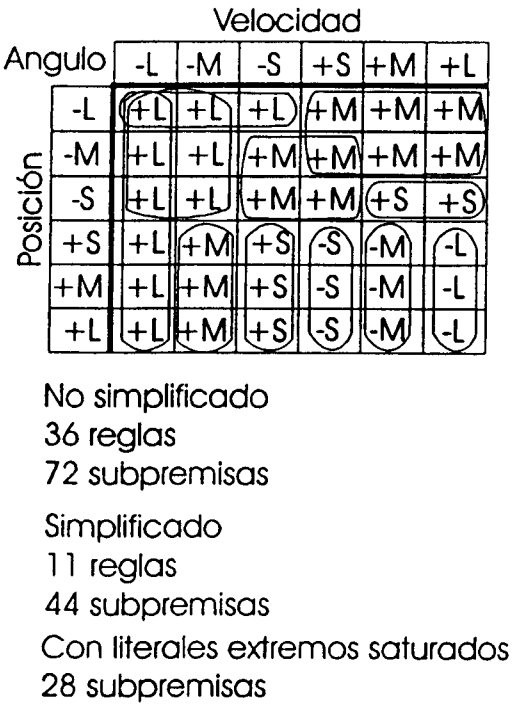

Figura 4. Ejemplo de aplicación de las simplificaciones a un sistema borroso

En la mayoría de los casos estudiados el número de subpremisas a examinar después de estas simplificaciones es del orden del 60% de las iniciales. Con una reducción todavía mayor en el número de reglas.

Aunque estas simplificaciones se dirigen a la realización de un procesamiento hardware, también son válidas para un procesamiento software siempre que se realice secuencialmente.

3 DISEÑO DEL CIRCUITO INTEGRADO

Teniendo en consideración las simplificaciones planteadas, hemos diseñado un circuito integrado para la ejecución del algoritmo máximo-mínimo-centro de área.

La arquitectura diseñada (figura 4) admite hasta 32 variables de entrada definidas con 8 bits. El cálculo del grado de activación de cada subpremisa (μ) se realiza mediante una LUT implementada en memoria ROM, válida para cualquier aplicación. El circuito integrado realiza el control global del sistema y el algoritmo de desborrosificación.

Este circuito trabaja secuencialmente en función de las instrucciones recibidas desde la memoria de programa. Estas instrucciones pueden definir subpremisas (formadas por la dirección de la variable Xi y los niveles Z e Y del literal), grados de activación de los actuadores y otras funciones de control.

El grado de segmentación con el que trabaja es de 2 niveles en el análisis del algoritmo máximo-mínimo y otros dos niveles para el cálculo del centro de área.. El divisor del cálculo del centro de área se realiza en 8 impulsos de reloj mediante un circuito de aproximaciones sucesivas.

El circuito se ha implementado en una FPGA tipo FLEX EPF8820 de Altera, consume 487 elementos lógicos. En comparación con otros controladores estudiados podemos destacar del circuito presentado:

a) Admite hasta 32 variables de entrada con precisión de 8 bits.

b) Puede controlar hasta 32 actuadores definidos con 8 bits.

c) El número máximo de subpremisas que determinan el sistema es de 30.000.

d) En un segundo puede analizar alrededor de 8.000.000 de subpremisas con un reloj de 10 MHz. Una velocidad notablemente superior al de un microprocesador y algo inferior a los controladores borrosos de procesamiento paralelo SIMD.

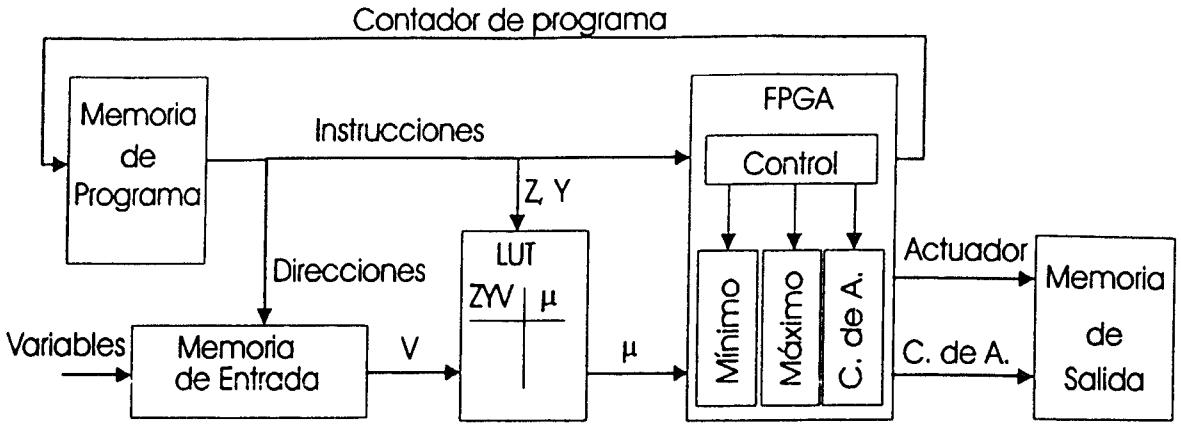

Figura 5. Arquitectura para inferencia borrosa secuencial segmentada mediante el algoritmo máximo-mínimo-centro de área.

Referencias

[1] H. J. Zimmermann, "Fuzzy Set Theory and Its Applications", Kluwer Academic Publishers, 1991.

[2] H. Eichfeld, M. Klimke, M. Menke, J. Nolles and T. Künemund, "A General-Purpose Fuzzy Inference Processor", IEEE Micro Magazine, vol. 15, nº 13, 1995.

[3] M. Togai and H. Watanabe, "Expert System on a Chip: An Engine for Real-Time Approximate Reasoning", IEEE Expert Syst. Mag., vol. 1, pp. 55-62, 1986.

[4] H. Watanabe, W. D. Dettloff and K. E. Yount, "A VLSI Fuzzy Logic Controller with Recon-figurable, Cascadable, Architecture", IEEE J. Solid State Circ., vol. 25, nº 2, pp. 376-381, 1990.

[5] V. Samoladas, L. Petrou, "Special-purpose Architectures for Fuzzy Logic Controllers", Microprocessing and Microprogramming, vol. 40, nº 4, pp. 275-289, 1994.

[6] D. L. Hung, "Dedicated Digital Fuzzy Hardware", IEEE Micro Mag., vol. 15, nº 4, pp. 31-39, 1995.

[7] L. de Salvador, J. Gutiérrez, "A Multilevel Systolic Approach for Fuzzy Inference Hardware", IEEE Micro Mag., vol. 15, nº 5, pp. 61-71, 1995.

[8] A. Kandel, "Fuzzy Hardware Challenges", IEEE Micro Mag., vol. 15, nº 6, pp. 61-67, 1995.

[9] G. Ascia, V. Catania, B. Giacalone, M. Russo, L. Vita, "Designing for Parallel Fuzzy Computing", resumen en IEEE Micro Mag., vol. 15, nº 6, pp. 62, artículo completo por internet pp. 1-11, 1995.

[10] A. Jaramillo-Botero, Y. Miyake, "Parallel, High-Speed PC Fuzzy Control", resumen en IEEE Micro Mag., vol. 15, nº 6, pp. 63, artículo completo por internet pp. 1-10, 1995.

[11] H. H. Li, N. Godfrey, Y. Ji, "A Beam-and-Ball Controller Prototype", resumen en IEEE Micro Mag., vol. 15, nº 6, pp. 61-67, artículo completo por internet pp. 1-7, 1995.

[12] A. Ruiz, J. Gutiérrez, J. A. F. Fernández, "A Fuzzy Controller with an Optimized Defuzzification Algoritm", resumen en IEEE Micro Mag., vol. 15, nº 6, pp. 61-67, artículo completo por internet pp. 1-10, 1995.

[13] W. Pedrycz, "Fuzzy Control and Fuzzy Systems", Ed. John Wiley & Sons, 1993.

[14] B. Kosko, "Neuronal Networks and Fuzzy Systems", Prentice-Hall International Editions, 1992.

[15] T. Terano, K. Asai, M. Sugeno, "Fuzzy System Theory & its Applications", Academic Press, 1995.

RICE: un sistema fuzzy en la conservación y estudio del comportamiento del arroz ensilado.

G. Buldain
NaSerSA*, UPNA
gbuldain@upna.es

P. Buldain
Univ. de Lérida

A.Lafarga
ITGA†

A. Pina
UPNA‡
pina@upna.es

F.J. Serón
C.P.S., UniZar
pseron@mcps.uniza

Resumen

Se presenta un sistema fuzzy de propósito general, orientado a usuario final y destinado al control y obtención de datos mediante reglas que suministra el experto. El desarrollo ha sido implantado satisfactoriamente por los especialistas del ITGA en la Cooperativa San Esteban (Arguedas, Navarra) para controlar y estudiar el comportamiento del arroz almacenado.

Palabras Clave: Control borroso, Sensores, Toma de decisiones .

1 Introducción

1.1 Descripción del problema a resolver

La conservación de grandes cantidades, Millones de kilos, de cereal ensilado durante periodos largos de tiempo es especialmente crítica en el arroz debido a que un control defectuoso ocasiona una rápida degradación de un producto que, por otro lado, está destinado fundamentalmente al consumo humano.

El problema se ve acentuado por la gran cantidad de variables que el experto utiliza en la toma de decisiones, tanto de parámetros físicos (actualmente 50, temperaturas, humedades, caudal de ventilación, costos energéticos, etc...) como de variables definidas por el usuario calculadas matemáticamente a partir de otras variables o deducidas por el experto mediante reglas (actualmente cerca de un centenar, focos de calor, perdidas por secado, máximas, medias, gradientes de temperatura, etc...). Muchas de estas variables plasman un conocimiento vago, impreciso, fruto de la experiencia como puede ser una previsión climática o un esperado precio de venta.

Parámetros como la humedad y la temperatura tienen una fuerte influencia en la conservación del grano, y su mantenimiento en un margen de seguridad es especialmente importante. Su control se realiza inyectando aire refrigerado o no, aprovechando gradientes de temperatura con el exterior. El aire inyectado producirá cambios en la temperatura y humedad, que pueden ocasionar perdidas económicas por secado o degradación del grano por humedad o temperaturas altas.

Las decisiones que se deben tomar deben garantizar la conservación, buscando un equilibrio entre el riesgo de degradación y las perdidas en peso, el coste de inyectar aire y otros criterios económicos.

1.2 ¿Por qué este sistema borroso?

Herramientas comerciales como pueden ser Fuzzitech de Microchip Technnology o la linea de productos borrosos de OMROM [FUZ94], [TRI94], [RENI95], [ECH95], estan destinadas fundamentalmente a a desarrolladores de hardware cerrado y no son adecuadas a esta clase de problemas, por el tipo de necesidades que plantean:

- Elevado número de variables y valores que intervienen en el proceso (mas de 100)
- Elevado número de reglas que el experto utiliza en el control (mas de 100)
- Necesidad de utilizar variables definidas por el usuario calculadas mediante funciones o reglas tanto para el control como para la obtención de resultados
- Debe ser una herramienta destinada a usuario, independiente en lo posible de la aplicacion a la que

*Navarra de Servicios, S.A.
†Instituto Técnico Agricola
‡Universidad Pública de Navarra
§Centro Politécnico Superior, Universidad de Zaragoza

lo destine

- No se puede separar la parte de desarrollo de la explotación. Se debe permitir al experto definir el comportamiento del sistema de forma autónoma ya que la complejidad del problema fuerza numerosos y continuos cambios

1.3 Características del sistema presentado

A continuación se resumen algunas de las características del sistema, y que han condicionado el desarrollo:

- Se ha dotado al usuario experto en una área de conocimiento de una herramienta de trabajo basada en lógica borrosa, independiente de la aplicación a la que la destina. Se ha buscado un equilibrio entre los recursos necesarios, complejidad de uso e implantación y conocimientos necesarios de lógica borrosa y otros ajenos a la especialidad que asegure que la herramienta sea necesaria y se use.

- Permite definir de forma sencilla un número no limitado de variables y reglas. Un analizador lexico y sintáctico valida la descripción que realiza el experto

- Permite que el experto defina variables cuyo valor se calcule mediante funciones (diferencia, máximo, media, etc...) o con reglas definidas por él. No existe diferencia entre variables físicas y variables definidas por el usuario. Tampoco existe diferencia entre variables de entrada y salida

- Monitorización permanente del proceso y estado de las variables

- Permite que el experto especifique que valores de variables hay que almacenar en disco para un posterior estudio del comportamiento y respuesta del grano

2 Componentes del sistema

El sistema está compuesto por dos unidades. Una componente fuzzy desarrollada para Windows en C++ y que actualmente se ejecuta en un PC 286 con 2Mb de RAM. Se comunica con una componente hardware basada en un microcontrolador Pic 1654 de Microchip, que recibe la información de las sondas y controla los actuadores. No es una aplicación dedicada y puede ejecutarse simultáneamente a otras aplicaciones.

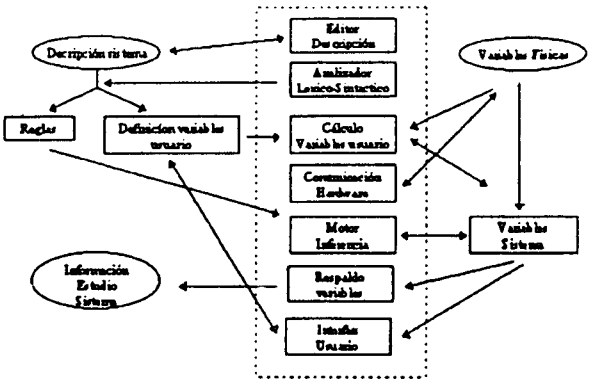

Figura 1: Descripción del componente Fuzzy.

2.1 Componente Fuzzy

2.1.1 Editor y analizadores

En la figura 1 se puede apreciar en primer lugar la existencia de un *editor y un analizador léxico y sintáctico* que, a partir de la descripción del sistema que realiza el experto en un lenguaje asociado a una gramática de contexto libre, obtiene la relación validada de reglas y definición de variables que intervienen almacenada en una estructura dinámica interna. El usuario dispone de la descripción de la gramática en formato BNF y diagramas sintácticos.

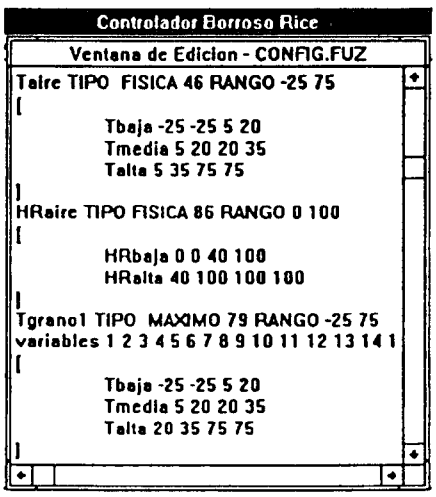

Figura 2: Edición del fichero de descripción del sistema.

2.1.2 Cálculo de variables de usuario

Este módulo obtiene los valores de las variables no físicas, utilizando la definición suministrada por el usuario y el estado actual de las variables. El valor de aquellas variables que se determina mediante reglas, será efectivo en el siguiente paso de inferencia.

Este
módulo gestiona también la activación/desactivación de variables. Cuándo el usuario desactiva una variable, se desactivan automáticamente y de forma recursiva las variables que se encuentran en el arbol de dependencias; si una variable es diferencia de dos variables, la desactivación de cualquiere de las dos provoca la desactivación de la dependiente. Si fuera una media o un máximo de varias variables, sería necesario deasactivar todas para desactivar la dependiente. Actualmente se está añadiendo la desactivación de variables mediante reglas, ya que sensores volumétricos determinan el nivel de llenado del silo, provocando la desactivación de aquellos sensores de temperatura que no estan dentro del grano. Consecuentemente se desactivarán también las reglas en las que intervengan variables desactivadas.

2.1.3 Motor de Inferencia

Funciones de pertenencia en la Borrosificación.

El sistema puede utilizar un número de variables y valores linguísticos limitado únicamente por los recursos del sistema. Actualmente, las funciones que utiliza el sistema son de tipo trapezoidal [MUJA94] [TRI94] [ZAD94], que se ajustan mas a la forma de trabajo del experto y que de momento las considera adecuadas. No hay que olvidar que el objetivo principal es obtener una herramienta orientada a usuario cuya incorporación a su ámbito de trabajo sea lo menos traumática posible. Una función se define con cuatro puntos α, β, γ y δ. Los grados de pertenencia de las variable se calculan mediante la siguiente fórmula:

$$\begin{cases} Si\ estamos\ en\ una\ zona\ de\ pendiente\ del\ trapecio \\ \mathcal{D} * \Delta, \\ Si\ estamos\ en\ una\ zona\ rectangular\ del\ trapecio \\ 1, \\ Si\ estamos\ fuera\ del\ trapecio \\ 0. \end{cases} \quad (1)$$

donde:

- Δ es la pendiente, $\frac{1}{\beta-\alpha}$ o $\frac{1}{\delta-\gamma}$
- $\mathcal{D}$ es o bien $valor_{entrada} - \alpha$ o bien $\delta - valor_{entrada}$ dependiendo de a que lado del trapecio estamos

Proceso de Inferencia.

El número de reglas solamente está limitado por los recursos del sistema y son de la forma:

$$\begin{cases} IF & \sum_{i=1}^{n} Entrada_i \quad IS \quad Valor_i \\ & THEN \\ & \sum_{j=1}^{m} Salida_j \quad IS \quad Valor_j \quad Peso \quad p \end{cases} \quad (2)$$

El sistema utiliza un motor clásico [DHR93] [VIO93] [RENI95]. Utiliza los operadores Min, Max y las reglas pueden tener asociado un peso. La activación/desactivación de reglas se realiza automáticamente y está guiada por las variables que intervienen en la misma. No se distingue entre variables de entrada o salida ni entre variables físicas o definidas por el usuario. La velocidad del paso de inferencia no es crítica para el tipo de aplicaciones a la que se destina esta herramienta y está garantizada fácilmente por la tecnología utilizada.

Método de desemborronamiento.

El sistema permite utilizar el método del centro de gravedad [RENI95], aunque el usuario de momento trabaja con valores monotónicos, quizás por la aplicación concreta para la que ha destinado el experto la herramienta.

2.1.4 Comunicación con el componente hardware

Este módulo solicita a través de un enlace serie a la unidad hardware (una caja), el valor de las variables físicas antes de cada paso de inferencia. Como la herramienta es independiente de la aplicación a la que se destina se realiza una asociación entre las variables definidas y los sensores en función de su orden de colocación, la primera variable que aparece se asocia al primer sensor, la segunda al segundo, y así sucesivamente. Así el usuario sabe que el sensor que coloca en la entrada 2 del componente hardware, y que mide por ejemplo la humedad exterior, está asociado con la segunda variable que ha colocado en la descripción y que ha llamado *Humedad Exterior*. Tras cada paso de inferencia envia el valor de las variables de salida a la unidad hardware para que esta active los diferentes actuadores. La asociación se realiza de forma análoga a como se realiza con las variables de entrada.

2.1.5 Respaldo de variables

Dado que estamos hablando de una herramienta destinada a usuario, era necesario dotar a la misma de la posibilidad de registrar los valores de las variables

periódicamente. Esta necesidad no está motivada únicamente para estudiar el comportamiento del sistema, reglas, valores linguísticos, etc..., sino también el comportamiento y evolución del grano almacenado, desarrollos bacterianos, focos de calor, etc... Mediante el interfaz de usuario el experto determina el periodo de registro en un archivo, y sobre que variables.

2.1.6 Interfaz de usuario

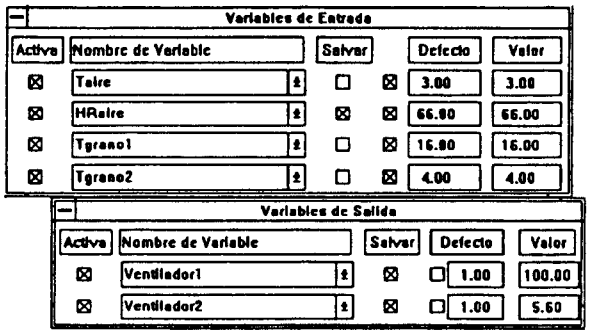

Figura 3: Monitorización del proceso.

Este módulo, permite la interacción del sistema con el usuario mediante un interfaz típico de Windows. El usuario puede visualizar los valores de las variables, activarlas, desactivarlas, colocar valores por defecto, etc... También puede editar la descripción del sistema y modificar los parámetros generales.

2.2 Componente Hardware

Este módulo está basado en un microcontrolador Pic [MDB95] [MDB95] que tiene un costo de pocos cientos de pesetas, con un programa desarrollado en ensamblador que intercambia paquetes a través de un enlace serie con el PC. El prototipo instalado permite incorporar sensores que transmiten su valor en forma de pulsos modulados en anchura, que nos servirá tras una interpolación para obtener el valor real de la variable. Los valores utilizados para la interpolación se almacenan en tablas para facilitar la calibración. Los actuadores que se utilizan actualmente son del tipo todo o nada, reles que controlan ventiladores y señales de alarma, no obstante el valor de las variables es continuo en un intervalo, que comprende los dos valores y la histéresis. Actualmente se utilizan 47 sensores de entrada, y 4 actuadores. En el prototipo instalado se ha tenido que asumir un sistema de sondas de temperatura ya existente que oculta en parte el carácter general de la herramienta.

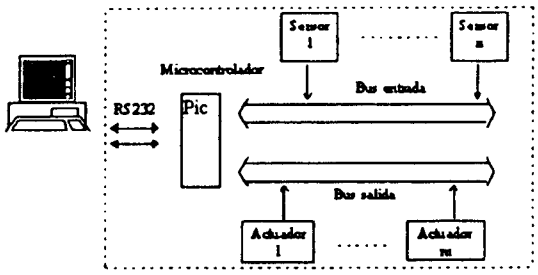

Figura 4: Descripción del Componente Hardware.

3 Conclusiones y trabajos futuros

Actualmente el prototipo está instalado, lleva varios meses controlando los silos y registrando los valores de las variables. Los expertos en cereal del ITG Agrícola, están estudiando y realizando gráficas de la evolución de los parámetros del grano almacenado.

En este momento se está incorporando la posibilidad de definir reglas que determinen el grado de activación de las variables y el peso de las reglas. Se van a incorporar sensores volumétricos de nivel de llenado, asociados a variables que intervendran en las reglas que desactivaran aquellos sensores que están en zonas del silo sin grano. Se ha observado que el experto utiliza la herramienta fuzzy para determinar mediante reglas el valor de algunas variables, como la eficacia del secado, y de hecho se estan incorporando otros tipos de sensores, como caudal de aire de ventilación, mas destinados al estudio o modelización del comportamiento del arroz que al control propiamente dicho.

Los técnicos del ITG Agrícola estan estudiando la posibilidad de implantar el sistema para el almacenamiento ensilado de otros cereales, aunque para estos los requisitos de control sean menos críticos.

Se está trabajando también en poder implantar todos los módulos, excepto el interfaz de usuario, en un microcontrolador para aquellas aplicaciones donde no se requiera una monitorización del proceso. La definición se realizaría en la aplicación actual, y posteriormente se transferiría al microcontrolador, con la consiguiente reducción de costos y espacio.

Bibliografía

[DHR93] Driankov D., Hellendoom H., Reinfrank M. : *An Introduction to Fuzzy Control.* Springer-Verlag, (1993).

[FUZ94] FuzzyTech MP Users Guide. 1994 Microchip.

[MDB95] Microchip Databook. 1995 Microchip.

[ECH95] Enbebed Control Handbook. 1995 Microchip.

[MUJA94] Munakata T., Jani Y.: Fuzzy Systems: an overview. *Communications of the ACM* Volume 37, Number 3 (March 1994) 69-76.

[TRI94] Enric Trillas, editor : *Fundamentos e introducción a la Ingeniería "fuzzy"*. Omron Electronics S.A., (1994).

[VIO93] Viot G. : *Creating a fuzzy-based inference engine*. Dr. Dobb's Journal, February 1993, 40-49.

[RENI95] Reyero R., Nicolás C.F. : *Sistemas de control basados en Lógica Borrosa : "Fuzzy Control"*. Omron Electronics S.A., Ikerlan, (1995).

[ZAD73] ZADEH L. A.: Outline of a New Approach to the Analysis of Complex Systems and decision Processes. *IEEE Trans. Syst., Man, Cybern.* Volume SMC-3, Number 1 (January 1973) 28-44.

[ZAD94] ZADEH L. A.: Fuzzy Logic, Neural networks, and soft Computing. *Communications of the ACM* Volume 37, Number 3 (March 1994) 77-84.

LOCALIZACIÓN Y CLASIFICACIÓN BORROSA DE DEFECTOS DE FABRICACIÓN EN PLANCHAS DE VIDRIO

L.A. Junco
Departamento de Matemáticas
ETSII de Gijón. Campus de Viesques.
33204 Gijón. Asturias
e-mail: junco@etsiig.uniovi.es

L. Sánchez
Departamento de Matemáticas
ETSII de Gijón. Campus de Viesques.
33204 Gijón. Asturias
e-mail: luciano@etsiig.uniovi.es

Resumen

En este trabajo se exponen los resultados preliminares obtenidos en la aplicación industrial de un clasificador borroso, cuyos parámetros han sido ajustados mediante algoritmos genéticos. Este sistema es empleado en el control de calidad de un proceso de fabricación de parabrisas de automóviles, por parte de una empresa cristalera asturiana.

Palabras Clave: Visión artificial, Enfoque, Clasificación borrosa, Algoritmos genéticos.

1 INTRODUCCIÓN

Hoy en día es cada vez más frecuente la utilización de Sistemas de Visión Artificial en la industria. La posibilidad de incorporar equipos de bajo coste a los procesos de inspección visual es un buen acicate para el desarrollo de nuevos productos.

El presente artículo trata precisamente de la automatización de una de las fases del control de calidad que se realiza en la industria de la fabricación del vidrio: la localización y clasificación de defectos presentes en las planchas de vidrio aparecidos durante el proceso de fabricación.

Los principales aspectos del vidrio examinados en los laboratorios son tres: su composición química, su cristalización y la aparición de defectos que puedan alterar la buena presencia del material o sus características técnicas. Este último control se desarrolla normalmente en la actualidad por un operario que examina visualmente (ayudado por un microscopio de pocos aumentos) una muestra del material. Como los defectos buscados pueden llegar a tener un tamaño de 50 micras, su inspección es lenta y laboriosa.

El sistema CVSIM, desarrollado en la Universidad de Oviedo, permite automatizar todo el proceso de busqueda y clasificación de defectos, de forma que no es necesaria la intervención de un operador. El clasificador, que está basado en conjuntos borrosos, se ha construido a partir de una descripción imprecisa de las características de los defectos, realizada por el propio operador. Los parámetros que definen cada etiqueta lingüística han sido obtenidos automáticamente, mediante un algoritmo de optimización genético.

2 SOLUCIÓN TÉCNICA ADOPTADA

El laboratorio de control de calidad de la empresa propuso como objetivo la localización de todos los defectos de tamaño superior a 50 micras, discriminando los defectos superficiales de los inmersos en el interior de la hoja de vidrio. Cada defecto se clasificará en una de las siguientes cuatro categorías: burbujas, manchas de estaño, piedras, y polvo o elemento extraño (ver Figura 2).

El sistema desarrollado para el análisis de cada muestra de cristal puede verse en la figura 1 . Se compone de un ordenador personal, una cámara monocromática (modelo CEMTYS CV-252C), un monitor y una mesa, donde va anclada la muestra y que proporciona a la cámara movimiento en tres ejes.

La mesa posee un controlador, que está conectado al ordenador a través de una linea serie. Una tarjeta (modelo DT2953-50Hz) incorporada al ordenador, digitaliza la señal enviada por la cámara, que es mostrada en todo momento a través del monitor. La medición de la profundidad se resolvió al dotar al sistema de movimiento perpendicular al plano de la muestra, con lo que puede realizarse un enfoque de la misma a distintas alturas.

Cada una de las imágenes adquiridas posee un tamaño de 512x512 pixels y 256 niveles de gris. El area cubierta por cada imagen es de 3x3 mm^2, de modo que un defecto de 50 micras de diámetro (el mínimo detectable) mide 12 pixels. Hemos estimado que este es el tamaño crítico para su reconocimiento.

Se procesan un total de 6.500 imágenes para cubrir completamente la superficie total de una mues-

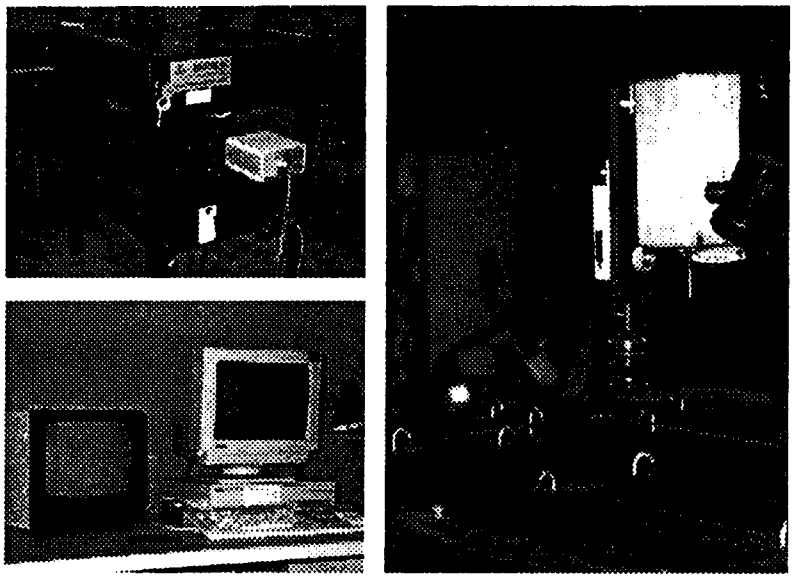

Figura 1: Sistema CVSIM

tra de cristal estándar (de 600x100 mm^2). Además, la medición de la profundidad conlleva la adquisición y análisis de dos nuevas imágenes por cada defecto. Dado el elevado número de imágenes que se están analizando, la velocidad del clasificador es muy importante.

3 EXTRACCIÓN DE INFORMACIÓN SOBRE LOS DEFECTOS A PARTIR DE UNA IMAGEN

Resumimos a continuación las técnicas de visión artificial empleadas para conseguir, a partir de una imagen, extraer la información que nos permite localizar y clasificar los defectos.

3.1 LOCALIZACIÓN DEL DEFECTO

El posicionamiento horizontal del defecto es inmediato con este montaje, ya que las coordenadas de la cámara son conocidas. La localización de cada uno de los defectos se realiza de modo simple, con una binarización y agrupamiento de los pixels más oscuros de cada imagen.

Pero el problema no es tan simple cuando se trata de determinar la situación del defecto dentro de la hoja de vidrio. Existen varias técnicas para medir la distancia a la que se encuentra un objeto de la cámara: las más usuales son la estereoscopía o la utilización de sensores especiales. Desde que Pentland [1] expuso su nuevo concepto de profundidad de campo como alternativa para el cálculo de la distancia basándose en el desenfoque, se han venido desarrollado numerosos trabajos que hacen uso de esta técnica. La base de estas aplicaciones estriba en el hecho de que un objeto situado lejos del plano de enfoque de la cámara aparece menos nítido. Esta falta de nitidez o borrosidad se puede medir y parametrizar en función de las características de la cámara, de la lente y de la distancia al objeto [2].

Varios autores [3] apuntan a la utilización de lentes motorizadas para conseguir mapas precisos de profundidad, con un rango amplio de distancias. En aquellas aplicaciones donde el rango de profundidades sea suficientemente estrecho, es posible (y presenta ciertas ventajas; cf. [4]) automatizar el movimiento de la cámara sin hacer variar el resto de parámetros.

Este último método se ajusta bien al perfil del sistema. La distancia se calcula a partir del grado de enfoque (medido mediante la función de Tennengrad [6]) de una serie de 3 imágenes del mismo defecto, enfocadas en planos diferentes. La imagen donde aparezca mejor enfocado cada defecto se tomará como base para la subsiguiente medición y clasificación.

3.2 EXTRACCIÓN DE CARACTERÍSTICAS

La extracción de características se reduce a un problema de descripción y análisis de una serie de regiones. Normalmente, la característica básica de una region es su forma; en nuestro caso, como todos los defectos salvo las burbujas presentan formas muy irregulares, basaremos el reconocimiento en otro tipo de características, que actuarán como descriptores de regiones. Hemos seleccionado 5 de estos descriptores: Contraste, momento de inercia[1], grado de agrupación, brillos y profundidad.

[1] Un descriptor basado en el momento de inercia, que mide el parecido de la figura con un anillo

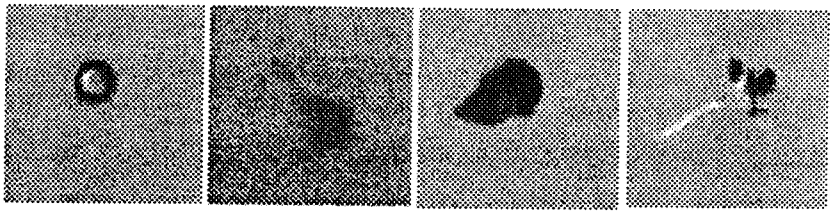

Figura 2: Defectos: Burbuja, estaño, piedra y mota de polvo

4 CONSTRUCCIÓN DEL CLASIFICADOR

En la construcción de este clasificador se unen dos circunstancias: por una parte, se cuenta con una descripción lingüística muy completa de los defectos (ver tabla 1). Por otra, el número de muestras de defectos disponibles para construir el clasificador es muy bajo, y algunos tipos de defectos están muy poco representados.

Se necesita un clasificador capaz de emplear toda la información lingüística disponible. Por esta razón, se optó por emplear reglas borrosas, en lugar de los algoritmos de clustering clásicos o redes neuronales. Para ahorrar tiempo de diseño, la elección de los parámetros que definen cada regla fue realizada con ayuda de un procedimiento de optimización genético.

Supongamos que hay N descriptores, cuyos dominios son conjuntos nítidos A_i, $i = 1 \ldots N$, y que las regiones de interés en cada uno de ellos pueden asociarse a conjuntos borrosos $\tilde{A}_i^j$, $j = 1 \ldots m_i$ (por ejemplo: contraste={alto, medio, bajo}) que forman una partición de A_i. Emplearemos la definición de la δ-ϵ partición de S. Montes [5],

$$(\tilde{A}_i^j \cap \tilde{A}_i^k)_\alpha = \phi \quad \forall j,k = 1,2 \ldots m_i, \quad \forall \alpha \geq \delta$$

$$(\bigcup_{j=1}^{m_i} \tilde{A}_i^j)_\alpha = A_i \quad \forall \alpha \leq 1 - \epsilon \qquad (1)$$

por ser la definición más flexible de las encontradas.

Supongamos además que las funciones de pertenencia de los $\tilde{A}_i^j$ dependen de p parámetros cada una

$$\tilde{A}_i^j(x) = f((a_i^j)^1, (a_i^j)^2 \ldots (a_i^j)^p, x), \quad x \in A_i. \qquad (2)$$

Caracterizaremos al tipo q-ésimo de defecto por una tupla $(q_1, q_2 \ldots, q_k, \ldots q_N)$, deducida del conocimiento del operador, con $q_k \in \{1 \ldots m_k\}$, y a cada defecto concreto x mediante N valores numéricos de los descriptores, $x = (x_1, x_2, \ldots x_i, \ldots x_N)$, $x_i \in A_i$. Asignaremos una confianza $\mu_q(x)$ a que el defecto x sea del tipo q-ésimo, $q = 1 \ldots N_c$, donde

$$\mu_q(x) = \tilde{A}_i^{q_1}(x_1) \wedge \tilde{A}_i^{q_2}(x_2) \wedge \ldots \wedge \tilde{A}_i^{q_N}(x_N) \qquad (3)$$

y, por último, definimos que el tipo $T(x) \in \{1 \ldots N_c\}$ del defecto caracterizado por x es aquel para el que

$$\mu_{T(x)}(x) = \max_{q=1 \ldots N_c} \mu_q(x). \qquad (4)$$

El clasificador queda completamente definido mediante los $2 + \sum_{i=1}^{N} pm_i$ parámetros que definen las etiquetas $\tilde{A}_i^j$ y los valores de δ y de ϵ. Para estimar estos parámetros empleamos un conjunto de ejemplos $X = \{x^1, x^2, \ldots x^{N_e}\}$, que hemos clasificado manualmente. Si sabemos que cada ejemplo x^k pertenece al tipo T_k, $k = 1 \ldots N_e$, la bondad del clasificador sobre X será mayor cuantos más efectos clasifique correctamente (aquéllos en los que $T(x^k) = T_k$), y a su vez el clasificador será tanto mejor si la confianza que se tiene en su clasificación (el valor $\mu_{T(x_k)}(x_k)$) es alta en los ejemplos correctamente clasificados, y baja en el caso contrario. Emplearemos entonces una expresión del tipo

$$E(X) = \sum_{\{x \in X : T(x) = T_k\}} \mu_k(x) - \sum_{\{x \in X : T(x) \neq T_k\}} \mu_k(x) \qquad (5)$$

considerando "ejemplos positivos" a los clasificados correctamente y "ejemplos negativos" a los incorrectos, de forma análoga al procedimiento seguido en [7]. Hemos empleado un algoritmo genético convencional (en cuyos detalles no entraremos) para maximizar la expresión $E(X)$ con respecto al conjunto de parámetros.

5 RESULTADOS EXPERIMENTALES

Tomando conjuntos borrosos trapezoidales, los $N = 5$ descriptores mencionados (contraste, momento de inercia, grado de agrupación, brillo y profundidad), $N_c = 5$ tipos de defectos, (burbujas, piedras en superficie, piedras en el interior, estaño, polvo), $N_e = 51$ ejemplos y dando valores iniciales aleatorios a los parámetros, se comprobó que las reglas elegidas eran coherentes, ya que en pocos cientos de generaciones se encontró una solución que clasificaba la práctica totalidad de los ejemplos. Este clasificador se implementó en el prototipo, y los resultados obtenidos durante los primeros dias de funcionamiento en la planta se resumen en la tabla 2, fila "Aciertos en test". La fila

Tabla 1: Características de los defectos

Defecto	Posición	Forma	Características
Burbuja	Interior	Anillo	Brillos Agrupado
Piedras en el interior	Interior	Poco Irregular	Alto contraste No brillos Agrupado
Piedras en la superficie	Superficie	Irregular	Brillos Opaco, con zonas más claras a su alrededor Agrupado
Estaño	Superficie	Irregular	Disgregado No hay brillos Contraste bajo
Polvo	Superficie	Irregular	Puede haber brillos Agrupado

Tabla 2: Porcentajes de clasificación de defectos

	Clasificador borroso	Perceptrón
Aciertos en entrenamiento	90.20%	88.23%
Aciertos en test	94.37%	77.46%

"Aciertos en entrenamiento" muestra los porcentajes de clasificación obtenidos en nuestro laboratorio con los ejemplos iniciales.

En esta misma tabla se muestran los resultados obtenidos al realizar la clasificación con un perceptrón multicapa. Obsérvese que, aunque los resultados obtenidos en el conjunto de entrenamiento son similares, los resultados en el conjunto de test son bastante mejores en el clasificador borroso. La aparente contradicción en el dato "aciertos en conjunto de test", que es superior al valor "aciertos en conjunto de entrenamiento", es debida a que las reglas de funcionamiento del clasificador borroso no fueron construidas a partir de los ejemplos.

6 CONCLUSIONES

Cuando se dispone de una buena definición lingüística de un clasificador y de un conjunto reducido de ejemplos, el clasificador borroso es una alternativa simple y robusta frente a otros sistemas, como las redes neuronales. En esta aplicación se han empleado algoritmos genéticos para estimar algunos parámetros del clasificador, obteniéndose unos resultados muy interesantes de cara a su aplicación práctica.

Agradecimientos

La investigación que ha dado lugar a este trabajo ha sido subvencionada por el contrato de investigación CN-95-091-B1 suscrito entre CRISTALERÍA ESPAÑOLA S.A. y la Universidad de Oviedo.

Referencias

[1] A.P. Pentland. (1987) "A new sense of depth of field". T-PAMI Vol 9, No.4, pp. 523-531

[2] M. Subbarao, T.C. Wei, G. Surya. (1994) "Focused Image Recovery from Two Defocused Images Recorded with different Camera Settings". CVPR(94), pp. 786-791

[3] M. Subbarao, T. Choi. (1995) "Accurate Recovery of Three-Dimensional Shape from Image Focus". T-PAMI Vol 17 No.3, pp. 266-274

[4] K. Nayar, Y. Nakagawa. (1994) "Shape from focus: An effective approach for rough surfaces". Int. Conf. on Robotics and Automation, pp. 218-225

[5] S. Montes, M.J. López-Palomo y P. Gil. (1995) "Las ϵ particiones difusas". ESTYLF 95

[6] G. Ligthart, F. Groen (1982) "A Comparison of different autofocus algorithms". Int. Conf. Pattern Recognition. pp 597-600.

[7] A. González y R. Pérez. (1993) "Un modelo experimental para determinar la estructura de una regla". Tercer Congreso español sobre tecnologías y lógica fuzzy. Santiago de Compostela. pp 9-16.

INFEDEC 2.1: SISTEMA DE AYUDA EN LA DECISIÓN DE DIAGNÓSTICO Y TRATAMIENTO DE ENFERMEDADES INFECCIOSAS

Ramiro Lago Bagüés
Director técnico de IBS.
C/ Juan Esplandiú, 13-7°, 28007-Madrid.

José Angel Olivas Varela
Dpto. Ingeniería Informática.
Universidad Antonio de Nebrija.
C/ Pirineos 55, 28040-Madrid.
E-mail: jolivas @dii.unnet.es

J. Pérez Fernández
Médico Adjunto del Centro de Salud "La Calzada", Gijón, Asturias.

C. Grávalos Castro
Médico Adjunto del Hospital "12 de Octubre", Madrid.

J. Alfonso Megido
Médico Adjunto de la UCI del Hospital "Valle del Nalón", Langreo, Asturias.

Resumen

Se presenta el sistema INFEDEC 2.1 como modelo de Sistema Basado en el Conocimiento aplicado al diagnóstico y tratamiento extrahospitalario en patología infecciosa de vías respiratorias superiores y otorrinolaringología. INFEDEC no es un programa comercial y se encuentra a la libre disposición del personal sanitario.

1 EL PROYECTO INFEDEC.

El proyecto INFEDEC ha contado con la financiación del Fondo de Investigación Sanitaria (proyecto 94/1913), dentro del plan nacional de I+D, y se ha desarrollado con el apoyo de la Unidad de Investigación del Hospital Valle del Nalón (Asturias). Para comprobar el correcto funcionamiento del programa, se ha puesto a prueba en los siguientes centros sanitarios: Hospital "Valle del Nalón" (Asturias), Centro de Salud "La Calzada" (Asturias), Hospital de Caridad "Jove" (Asturias).

El núcleo del proyecto INFEDEC es la realización de un sistema computacional que ayude a los médicos de atención primaria en estos dos aspectos:

- Gestión de la información (control de historiales clínicos, examen estadísticos sobre epidemiología, etc.).
- Ayuda a la decisión médica, haciendo que la computadora simule el razonamiento y modo de solucionar los problemas de un profesional experimentado.

De esta forma el ordenador no es sólo un administrador de información, sino que además ayuda, mediante un conocimiento actualizado, en las decisiones clínicas. Otro aspecto importante es que de esta forma se consigue que los estudiantes en medicina realicen un aprendizaje interactivo, ya que pueden aprender mediante casos y obtener una respuesta razonada y justificada. Con INFEDEC se permite acceder a una enseñanza interactiva de estas enfermedades, ya que el ordenador permite contrastar decisiones diagnósticas y terapéuticas, además de asociar dichas decisiones con un texto descriptivo de la patología infecciosa. Las tareas fundamentales que realiza el programa aparecen en la figura 1:

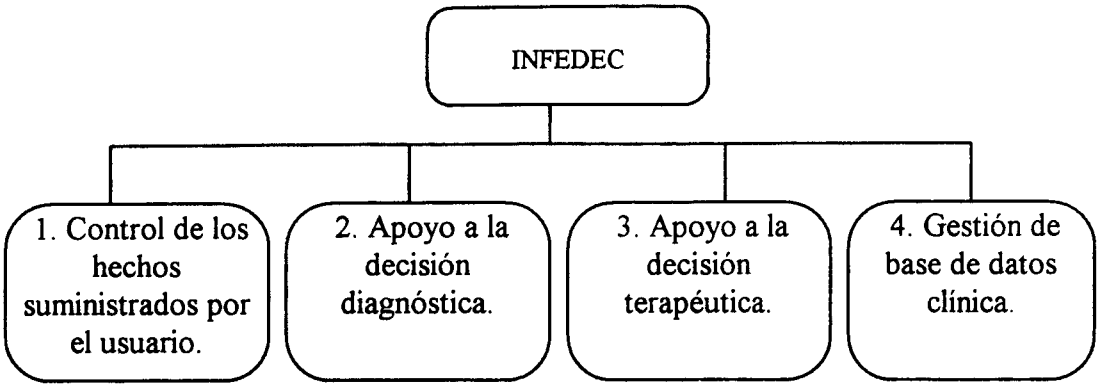

Figura 1: Tareas fundamentales del programa.

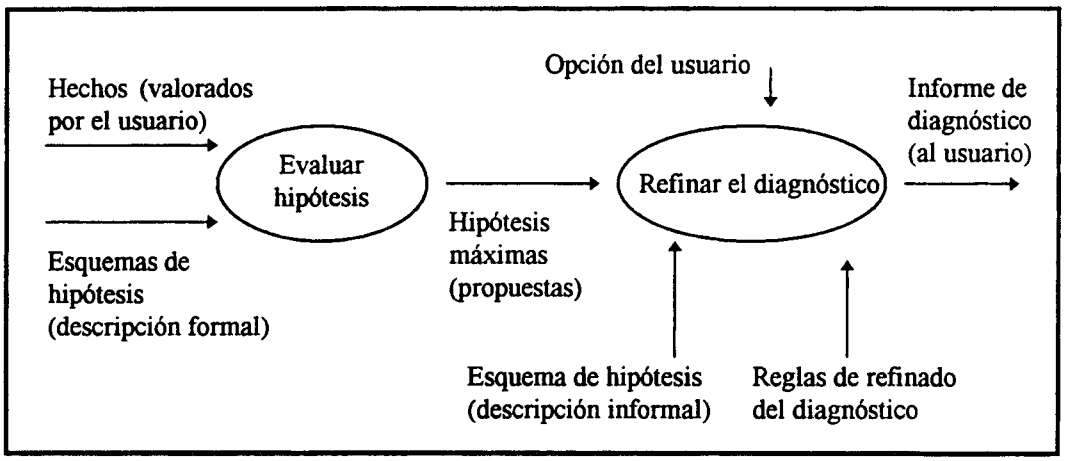

Figura 2: Proceso global.

2 MODELO LÓGICO.

Los pasos a seguir en el proceso de diagnóstico son básicamente los siguientes (Figura 2):

1. Evaluación de las hipótesis: obtenemos una lista de hipótesis propuestas, ordenadas por su grado de credibilidad.
2. Refinamiento del diagnóstico. Se seleccionan aquellas que han superado cierto umbral de aceptabilidad y se les añade la información textual (comentario, descripción, etc.). El usuario puede optar por una de las propuestas o por otra no presentada por el sistema.

El núcleo del proceso de diagnóstico se encuentra en el punto primero, por ello, nos centraremos en él. El razonamiento médico, al igual que el razonamiento científico y técnico sigue una lógica de conjeturas y refutaciones; es decir, el médico desde su encuentro con el paciente, comienza a emitir hipótesis diagnósticas que enseguida empiezan a ser contrastadas, estas hipótesis pueden ser verificadas o, por el contrario, ser refutadas. En el caso de ser refutadas, es decir, que no resistan la contrastación experimental; entonces son sustituidas por otras que se adapten mejor a los hechos. Este proceso no es esencialmente diferente del método de conjeturas y refutaciones expresado por Popper como un modelo de resolución de problemas.

En este programa se representa este proceso asignando a cada hipótesis:

- Grado de confirmación, que nos indica la medida en que los hechos (o también otras hipótesis) apoyan la hipótesis que estamos examinando.
- Grado de falsación, que representa la medida en que los hechos (o también otras hipótesis) refutan la hipótesis.

El resultado de combinar ambos factores es el grado de credibilidad de dicha hipótesis. Por tanto cada hipótesis representada por el programa incluye, entre otras cosas, un conjunto de hechos que la falsifican o confirman.

El modelo lógico para representar los factores de confirmación, falsación y credibilidad ha sido la lógica borrosa junto con la teoría de prototipos de Zadeh.

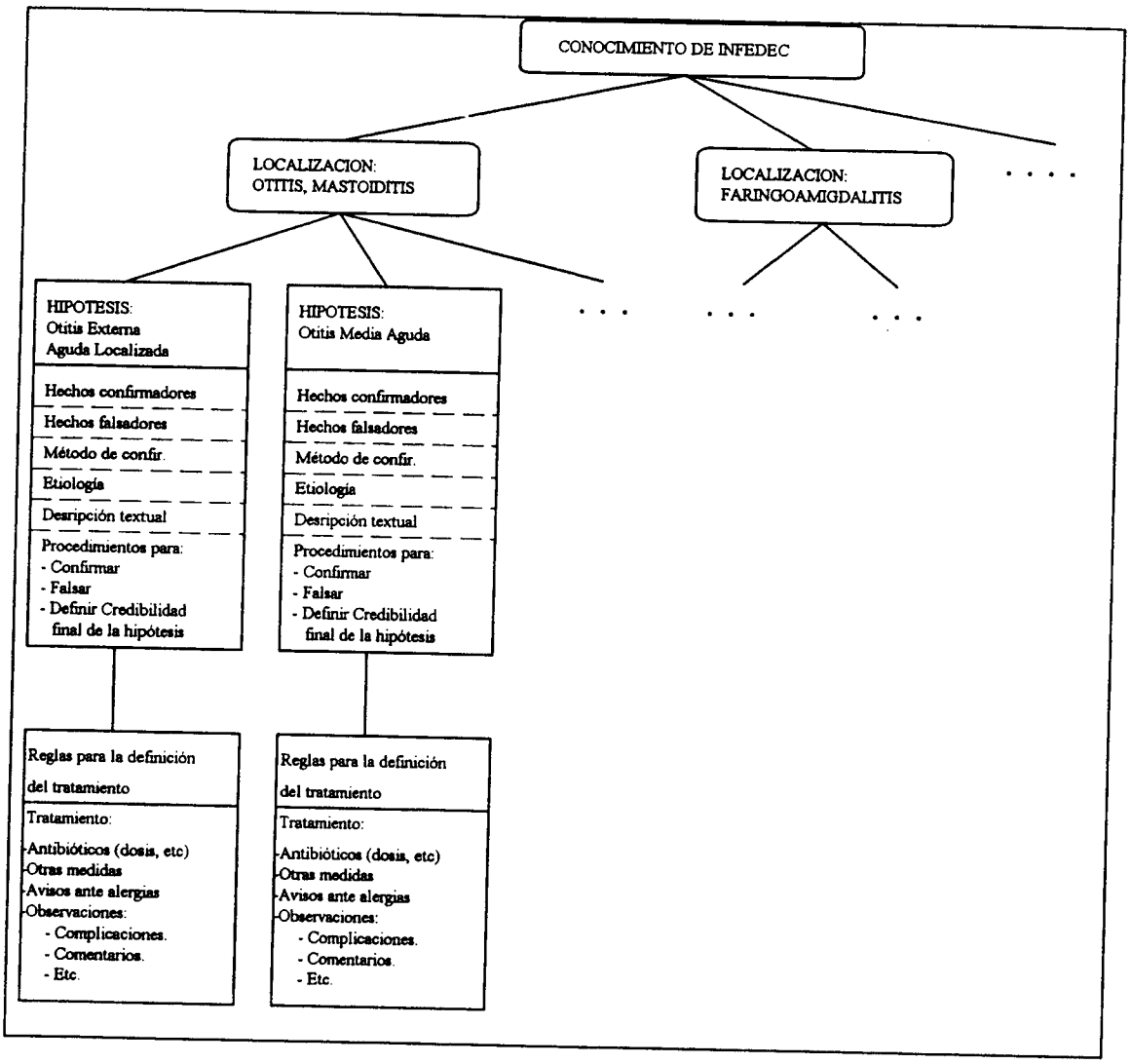

Figura 3: Conocimiento en INFEDEC.

3 FUNCIONALIDADES DE INFEDEC

3.1 FILIACIÓN DEL PACIENTE

INFEDEC permite introducir los datos de filiación del paciente que luego pueden ser almacenados en la **base de datos**.

3.2 DIAGNÓSTICO

A partir de los hechos introducidos por el usuario, INFEDEC presenta unas **propuestas de diagnóstico de sospecha**, ordenadas por su grado de **credibilidad**. Cada propuesta de INFEDEC está **justificada**: ya que se informa de las razones (hechos) que justifican dicha propuesta y se hace referencia a **bibliografía**. Además cada propuesta tiene una **descripción** textual que ayuda al usuario a tener una visión más completa tanto del diagnóstico como del tratamiento de dicha infección. Además informa de los **microorganismos** que pueden producir la infección, así como del **método(s) de confirmación** del diagnóstico. Se permite al usuario optar por una alternativa que no hubiera sido propuesta por el programa. Un ejemplo de propuestas de diagnóstico aparece en la figura 4:

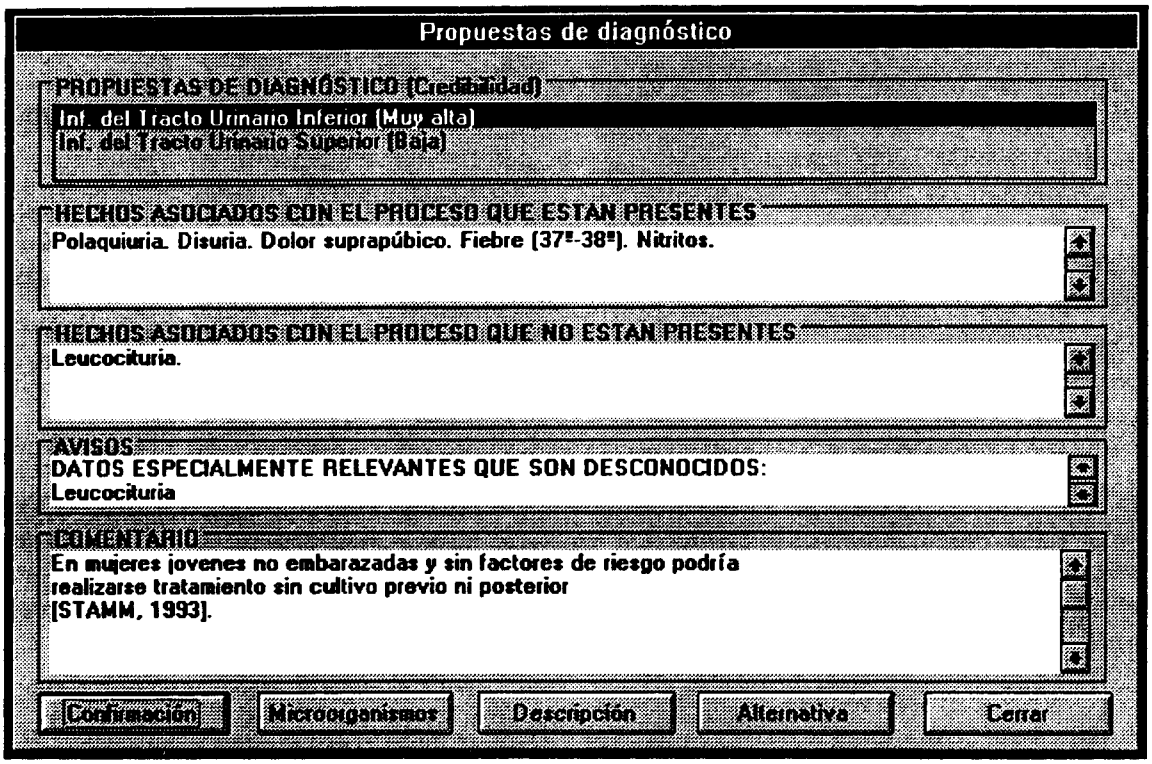

Figura 4: Ejemplo de propuestas de diagnóstico.

3.3 TRATAMIENTO

A partir del diagnóstico INFEDEC presenta unas **propuestas de tratamiento empírico**, ordenadas por su grado de adecuación (preferencia). Para cada propuesta se especifica las **características de administración,** sin presentar marcas comerciales. Además se muestra **otras posibles medidas**, así como una serie de **observaciones**. El usuario puede anotar una alternativa que no haya sido propuesta por el programa. Un ejemplo se puede ver en la figura 5:

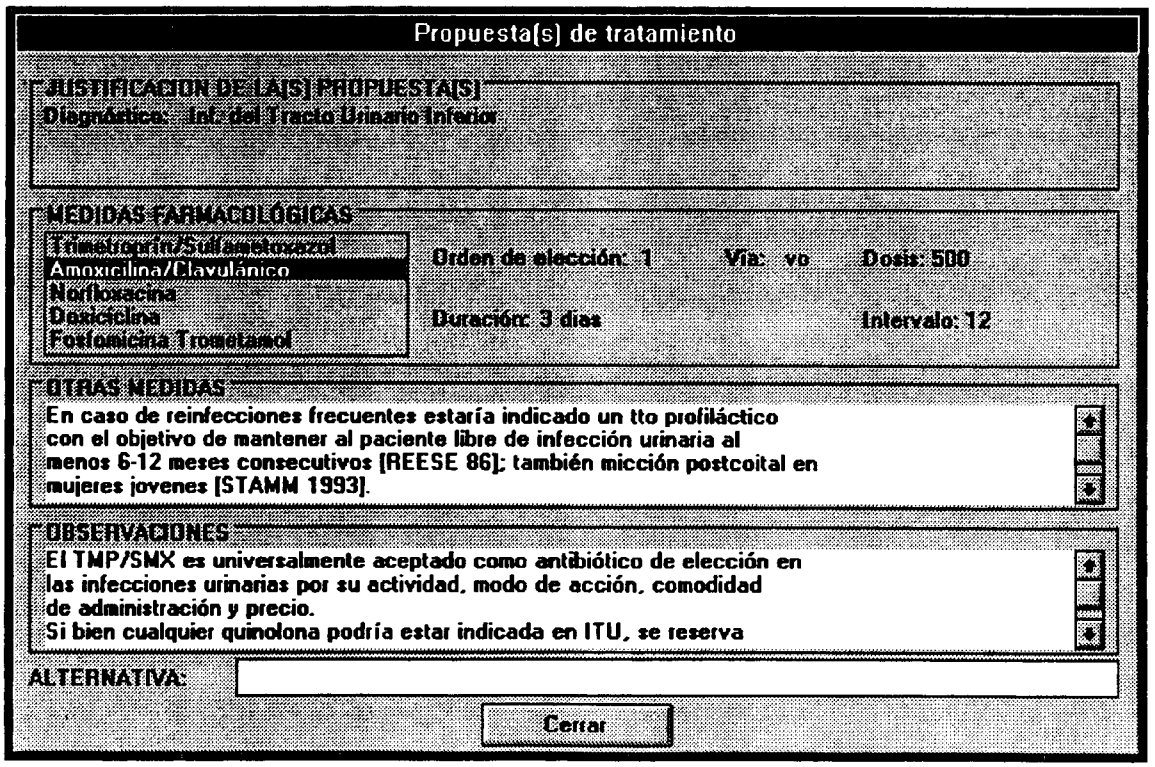

Figura 5: Ejemplo de tratamiento.

3.4 VADEMÉCUM

Se presenta un vademécum adaptado al ámbito del programa (infecciones respiratorias y del tracto urinario).

3.5 BIBLIOGRAFÍA

Se ofrece también una lista de referencias bibliográficas utilizadas.

3.6 BASE DE DATOS

Una vez que se haya(n) realizado la(s) decisión(es), se pueden salvar en la **base de datos de actos asistenciales**, incluyendo datos de **filiación** del paciente. Sobre la base de datos se pueden realizar consultas. A partir de la base de datos el programa nos muestra la distribución **estadística** de los pacientes de acuerdo con dos factores: sexo y edad.

4 REQUERIMIENTOS HARDWARE

Los requisitos hardware son un ordenador 386 o superior con al menos 2 Mb. de RAM y 5 de Disco Duro. INFEDEC se ejecuta sobre sistema Windows 3.0 o superior.

Referencias

[1] Olivas, J. A., Lago, R. y Sobrino, A. (1992). Una posible aplicación de la teoría de prototipos de Zadeh a la gestión de la evolución de la siniestralidad de los incendios forestales en un motor de inferencia. Actas del II Congreso Español sobre Tecnologías y Lógica Fuzzy; Facultad de Informática, Universidad Politécnica de Madrid, pp. 47-57.

[2] Zadeh, L. A. (1982). A note on prototype set theory and fuzzy sets. Cognition, 12, 291-297.

UN PROTOTIPO DE SISTEMA EXPERTO DIFUSO PARA LA VALORACION VISUAL DE PAISAJES

R. Martínez-Béjar
CSIC - CEBAS
Avda. La Fama, 1
30080 - Murcia
e-mail: rodrigo@natura.cebas.csic.es

J.M. Cádenas
Dpto. Informática y Sistemas
Universidad de Murcia
Campus de Espinardo,
30071 - Espinardo (Murcia)
e-mail:jcadenas@dif.um.es

F.Martín
Dpto. Informática y Sistemas
Universidad de Murcia
Campus de Espinardo
30071 - Espinardo (Murcia)
e-mail: fmartin@dif.um.es

Resumen

En este trabajo, se ha diseñado un prototipo de sistema experto para ayudar en tareas de valoración paisajística. Haciendo uso de la lógica difusa, se han seleccionado modelos capaces de manipular variables lingüísticas, así como mecanismos de inferencia que han resultado de gran ayuda para la actividad experta entorno al estudio de los paisajes caracterizada, entre otras cosas, por el frecuente uso de términos vagos e imprecisos. En concreto, se ha escogido el Modus Ponens generalizado (GMP) como instrumento básico de inferencia, mostrándose luego su eficacia para un ejemplo concreto.

1 INTRODUCCIÓN

Aunque los sistemas basados en conocimiento vienen siendo ampliamente aplicados a tareas de planificación medioambiental desde comienzo de la década de los ochenta [10,3,4,5], dichos sistemas han tenido a penas impacto en tareas de planificación física, en parte debido a la necesidad de que los expertos han de considerar infinidad de factores. En particular, ningún sistema basado en conocimiento ha sido desarrollado hasta la fecha para abordar tareas propias de valoración visual paisajística.

La valoración visual de un paisaje es básicamente una tarea de clasificación que requiere gran experiencia. Además, se ha constatado que los expertos en dicha tarea introducen imprecisión en el uso de etiquetas linguísticas que describen su razonamiento utilizado en el transcurso de su actividad cotidiana. Concretamente, ellos formulan hipótesis a partir de características difíciles de expresar cuantitativamente (por ejemplo, *naturalidad alta*) , para lo que la Teoría de conjuntos difusos resulta, sin embargo, útil.

Por otro lado, la Lógica Difusa permite modelar situaciones como la descrita a través de atributos difusos, posibilitando, además, un *calculus* que maneja cualidades.
En este trabajo, se ha diseñado un prototipo de sistema experto para valorar paisajes cuya descripción se hace, en general, a través de etiquetas linguísticas a las que, con frecuencia, se le añaden modificadores. Actualmente se trabaja en la fase de implementación de dicho prototipo para, posteriormente, realizar un estudio de validación de resultados tanto desde el punto de vista de la eficacia de dicho sistema, como de su presentación.

La sección 2 de este trabajo proporciona información básica sobre la tarea de la valoración visual de paisajes. En la sección 3 se muestra cómo la Teoría de conjuntos difusos puede utilizarse en este problema. La siguiente sección contiene el tipo de funciones de pertenencia utilizadas en este trabajo así como el mecanismo de inferencia usado. Finalmente, en la sección 5 se pone de manifiesto la eficacia de dicho mecanismo a través de un ejemplo.

2 CARACTERISTICAS GENERALES DEL DOMINIO DE APLICACIÓN

El objetivo de la valoración visual de un paisaje consiste en obtener el vector B = (calidad visual, fragilidad visual), donde la *calidad visual* es aquí sinónimo de belleza o valor estético, mientras que la *fragilidad visual* se puede definir como la susceptibilidad de un paisaje al cambio cuando se desarrolla un uso sobre él, expresando el grado de deterioro que el paisaje experimentaría ante la incidencia de determinadas actuaciones [9]. Si los valores de las variables usadas para describir las características visuales las determinan expertos situados en el interior de cada zona a ser valorada, se añade el

término *intrínseca* a cada uno de los elementos de B, que se puede escribir abreviadamente como B = (CVI, FVI).

Para calcular los valores del vector B, los expertos entrevistados descomponen la tarea de estudio del paisaje en otras tres subtareas, a saber, I_v, I_h e I_g, que consisten, respectivamente en determinar B según las características de vegetación y usos, las características hídricas y las geológico-geomorfológicas. El presente trabajo se ha limitado a la subtarea I_v.

Así pues, como se pone de manifiesto en [8], puede decirse que la valoración visual de un paisaje es un ejemplo típico de clasificación en el contexto de un proceso más global de interpretación y catalogación, llamado *planificación física*, en el que confluye conocimiento heterogéneo, surgiendo, por tanto, ciertas dificultades funcionales inherentes a todo estudio interdisciplinar.

Por otro lado, para definir los criterios de clasificación previamente elicitados, se hizo uso de un conjunto de más de 40 reglas de producción tales como "*IF la densidad de vegetación es alta AND la especie predominante es el pino AND los pinos no son jóvenes AND la zona es estratificada AND la naturalidad de la vegetación es alta THEN la calidad visual intrínseca de la zona es muy alta*".

Aunque esta regla fué contrastada formalmente con los expertos participantes en este proyecto, se puede decir que aquella posee ciertos matices difíciles de formalizar. Así, por ejemplo, cuando ante un estudio real llevado a cabo por varios expertos, se les preguntó que si la naturalidad de la vegetación era alta, cada uno de ellos, en general, respondía de distinta forma, ya que el concepto de naturalidad alta no está expresado de una forma cuantitativa precisa.

Además, algunos de los expertos consultados respondieron a dicha cuestión añadiendo adverbios de distintos tipos (modales, de frecuencia, de cantidad, etc). En otras palabras, se trata de un dominio caracterizado por la imprecisión y la vaguedad, por lo cual, la lógica difusa resulta adecuada para abordar este tipo de aserciones cualitativas [7]. A continuación, se muestra la idoneidad de usar la teoría de conjuntos difusos en el problema que acaba de presentarse.

3 TRATAMIENTO FUZZY DEL PROBLEMA

De acuerdo a [2], un sistema experto difuso está compuesto por reglas que pueden operar con etiquetas linguísticas, conjuntos difusos y relaciones, de modo que resulta factible procesar los datos de una determinada base de conocimiento y proporcionar una salida apropiada correspondiente a una particular entrada difusa. Dicho de otro modo, todo sistema experto difuso se caracteriza por lo siguiente:

1. El sistema puede manejar datos difusos.
2. Su entrada está compuesta por atributos imprecisos, los cuales pueden estar representados, bien por etiquetas linguísticas difusas, o bien por conjuntos difusos discretos.
3. Las reglas se definen de manera que puedan operar con datos difusos.
4. El resultado final es un conjunto difuso.

Por otra parte, los expertos deseaban que el sistema, además de tener capacidad para tratar información imprecisa del tipo *la densidad de vegetación es alta* (estando *alta* representado por un conjunto difuso) , pudiera admitir la posibilidad de que un modificador, por ejemplo el término *muy*, transformara *alta* en un conjunto difuso del mismo tipo. De este modo, se podrían admitir sentencias tales como *la densidad de vegetación es muy alta*

Para resolver este problema, se puede hacer uso de la función TFM (ver [6]), según la cual, si **D** es un conjunto difuso y **m** un modificador de verdad, el resultado de la modificación de **D** por medio de **m**, escrito TFM(D/m), es un conjunto difuso **mD** tal que $\mu_{mD}(x) = \mu_m(D(x))$, donde la función $\mu_S(v)$ representa el grado de pertenencia de v con respecto al conjunto S.

En el siguiente epígrafe, se presentan el tipo de funciones de pertenencia utilizadas para representar adjetivos y modificadores difusos, así como el sistema de inferencia que trata con reglas difusas.

4 MODELACIÓN DEL SISTEMA DE RAZONAMIENTO DIFUSO

Antes de elegir un sistema de inferencia apropiado para este problema, se modelaron mediante variables difusas aquellas etiquetas linguísticas cuyos conjuntos de valores responden al formato {bajo, medio, alto}, {pequeño, medio, grande}, etc.

Supongamos que I(x), M(x) y D(x) representan las funciones de pertenencia de las respectivas variables difusas asociadas a las etiquetas linguísticas del tipo mencionado en el párrafo anterior. En estas condiciones, estas etiquetas linguísticas pueden modelarse por medio de los siguientes conjuntos difusos [12].

$$I(x) = \begin{cases} 1 \text{ si } x \le \alpha \\ 1 - 2\left(\frac{x-\alpha}{\gamma-\alpha}\right)^2 \text{ si } \alpha \le x \le \beta \\ 2\left(\frac{x-\gamma}{\gamma-\alpha}\right)^2 \text{ si } \beta \le x \le \gamma \\ 0 \text{ si } x \ge \gamma \end{cases}$$

$$D(x) = \begin{cases} 0 \text{ si } x \le \alpha \\ 2\left(\frac{x-\alpha}{\gamma-\alpha}\right)^2 \text{ si } \alpha \le x \le \beta \\ 1 - 2\left(\frac{x-\gamma}{\gamma-\alpha}\right)^2 \text{ si } \beta \le x \le \gamma \\ 1 \text{ si } x \ge \gamma \end{cases}$$

$$M(x) = \begin{cases} 0 \text{ si } x \le \alpha \\ 2\left(\frac{x-\alpha}{\beta-\alpha}\right)^2 \text{ si } \alpha \le x \le \frac{\alpha+\beta}{2} \\ 1 - 2\left(\frac{x-\beta}{\beta-\alpha}\right)^2 \text{ si } \frac{\alpha+\beta}{2} \le x \le \beta \\ 1 - 2\left(\frac{x-\beta}{\gamma-\beta}\right)^2 \text{ si } \beta \le x \le \frac{\beta+\gamma}{2} \\ 2\left(\frac{x-\gamma}{\gamma-\beta}\right)^2 \text{ si } \frac{\beta+\gamma}{2} \le x \le \gamma \\ 0 \text{ si } x \ge \gamma \end{cases}$$

Así, por ejemplo, el conjunto difuso PINAR_DENSO puede representarse por medio de las funciones I(x), M(x) y D(x), con $\alpha = 70$, $\beta = 80$ y $\gamma = 90$

El resto de predicados difusos pueden tratarse a través de la ya mencionada función TFM de la siguiente manera: $\mu_{mD}(x) = [\mu_D(x)]^\alpha$ de modo que

$0 < \alpha < 1$ si el modificador m indica que D no se cumple con claridad. Ejemplos típicos serían *más o menos, casi* o *poco*;

$\alpha > 1$ si m indica claramente que D se cumple en grado acusado. Es el caso de modificadores como *muy, mucho, muchísimo* o *extremadamente*.

Por ejemplo, si se tiene la función difusa MATORRAL_BAJO(x): [0,200] → [0,1] (con la altura en cms.) y si se escoge una función polinomial para modelar la función de pertenencia $\mu_{matorral\_bajo}(x)$, por ejemplo, la función $\left(1 - \frac{x}{200}\right)^{1/2}$, entonces, en general, se puede escribir $\mu_{modificador\ matorral\_bajo}(x) = \left(1 - \frac{x}{200}\right)^{\alpha/2}$.

Por otra parte, el sistema propuesto manipula reglas cuya estructura responde al esquema

IF $(X_1$ is $V_1) \wedge (X_2$ is $V_2) \wedge \ldots\ldots \wedge (X_n$ is $V_n)$

THEN Y is A, donde V_i , i = 1, 2, .., n, y A son subconjuntos de los conjuntos C_{vi} y C_A, respectivamente.

Por otra parte, se supone que la información que aporta el usuario al sistema posee el siguiente formato:

$(X_1$ is $W_1) \wedge (X_2$ is $W_2) \wedge \ldots\ldots \wedge (X_n$ is $W_n)$

donde W_i , i = 1, 2, .., n, modifica V_i .

El mecanismo de inferencia descansa básicamente sobre el Modus Ponens Generalizado (GMP), el cual, permite deducir Y is A' , [1]. Además, [11] ha encontrado una forma de evaluar A', a través de funciones de distribución de posiblidad, como sigue:

$\mu_{A'}(y) = \sup_{\{xi\}} \{\min [P_1(x_1, x_2, \ldots, x_n), P_2(x_1, x_2, \ldots, x_n, y)]\}$, donde

$P_1(x_1, x_2, \ldots, x_n) = \min [\mu_{W1}(x_1), \mu_{W2}(x_2), \ldots, \mu_{Wn}(x_n)]$;

$P_2(x_1, x_2, \ldots, x_n, y) = \min [1, 1 - H(x_1, x_1, \ldots, x_n) + \mu_A(y)]$;

$H(x_1, x_2, \ldots, x_n) = \min [\mu_{v1}(x_1), \mu_{v2}(x_2), \ldots, \mu_{vn}(x_n)]$.

En la siguiente sección, se aplica el sistema de inferencia que se acaba de explicar a un ejemplo concreto de valoración visual de un paisaje.

5 EJEMPLO

Supongamos que en la base de conocimiento está la siguiente regla: If *la zona bajo estudio está dominada por matorral bajo y pino joven* then A = [0.7/CVI alta, 0.3/CVI muy alta](CVI representa la capacidad visual intrínseca asociada a la zona en cuestión), mientras que el usuario tiene la evidencia de que *la zona bajo estudio está dominada por matorral poco bajo y pino muy joven.*

Además, se supondrá que los correspondientes subconjuntos difusos (proporcionados por los expertos) son los siguientes:

matorral_bajo = [1/10, 0.7/50, 0.5/100, 0.1/150, 0/200];

pino_joven = [1/0, 0.99/1, 0.9/3, 0.85/10, 0.7/20, 0.4/30, 0.2/50, 0.1/70, 0/100].

Si a continuación se aplica la función TFM para el caso de funciones de pertenencia del tipo

$$\mu_{\text{etiqueta\_linguistica}}(x) = \left(1 - \frac{x}{x_{max} - x_{min}}\right)^{1/2},$$

donde etiqueta_linguística ∈ {*matorral bajo, pino joven*};

$$x_{max} = \begin{cases} 200 \text{ si etiqueta\_linguística} = matorral\_bajo \\ 100 \text{ si etiqueta\_linguística} = pino\_joven \end{cases}$$

$$x_{min} = \begin{cases} 10 \text{ si etiqueta\_linguística} = matorral\_bajo \\ 0 \text{ si etiqueta\_linguística} = pino\_joven \end{cases}$$

se obtiene que $\mu_{matorral\_poco\_bajo}(x) = (\mu_{matorral\_bajo}(x))^{\alpha 1}$, $0 < \alpha_1 < 1$; $\mu_{pino\_muy\_joven}(x) = (\mu_{pino\_joven}(x))^{\alpha 2}$, $\alpha_2 > 1$. Para este ejemplo, se ha valorado α_1 y α_2 como 0.9 y 1.5, respectivamente. Si ahora se aplica la función TFM, se llega a que *matorral_poco_bajo* = [1/10, 0.89/50, 0.76/100, 0.57/150, 0/200] y *pino_muy_joven* = [1/0, 0.99/1, 0.97/3, 0.92/10, 0.84/20, 0.76/30, 0.59/50, 0.40/70].

Finalmente, aplicando las fórmulas de Yager, se obtiene que $\mu_{A'}$(CVI alta) = 0.89 y $\mu_{A'}$(CVI muy alta) = 0.76.

6 CONCLUSIONES

El sistema diseñado, al menos inicialmente, se ha concebido para el dominio de la valoración paisajística. Sin embargo, puede ser utilizado en otros dominios de aplicación con características similares, es decir, en tareas de clasificación, además de los propios de planificación medioambiental.

En este sentido, fueron los expertos participantes en este proyecto quienes decidieron que, ante la complejidad subyacente a la tarea de valorar un paisaje, en relación al resto de tareas que ellos realizan habitualmente, un sistema experto capaz de responder a las características antes esbozadas sería asimismo susceptible de ser utilizado con éxito en otras tareas de planificación medioambiental, en particular, en planificación física.

Aunque, por el momento, el prototipo se halla en su fase de implementación, si el sistema se usa como sistema de ayuda a la decisión, ello conlleva un aumento de velocidad en el proceso de análisis del paisaje, cuando menos. Además, permite formular teorías en base a información vaga e imprecisa.

En definitiva, este sistema supone una innovación que puede ser muy útil, al menos en ciertos procesos de planificación medioambiental que requieran tareas de clasificación complejas en cuanto al número y naturaleza de las variables consideradas, así como a los valores, que pueden ser vagos e/o imprecisos, como es el caso de la valoración visual de un paisaje.

Referencias

[1] J. F. Baldwin, and B. W Pilsworth (1980). Axiomatic approach to implication to approximate reasoning with fuzzy logic. *Fuzzy Sets and Systems* **3**: 193 - 219.

[2] J. J. Buckley, W. Siler, and D. Tucker (1986). A fuzzy expert system. *Fuzzy Sets and Systems* **20**: 1 - 16.

[3] J. R. Davis, and P. M. Nanninga (1985). GEOMYCIN: Towards a geographic expert system for resource management. *J. Environ. Mgmt* **20**: 377-390.

[4] J. R. Davis, J. R. L. Hoare, and P. M. Nanninga (1986). Developing a fire management expert system for Kakadu National Park, Australia. *J. Environ. Mgmt* **22**: 215-227.

[5] R. Hunt, D. A. J. Middleton, J. P. Grime, and J. G. Hodgson (1991). TRISTAR: an expert system for vegetation processes. *Expert Systems* **8**: 219-216.

[6] A. Kandel (1990). *Fuzzy mathematical techniques with applications*, Reading, MA, Addison-Wesley.

[7] R. Lopez-De Mantaras (1990). *Approximate Reasoning Models*, Ellis Horwood Limited, Chichester, West Sussex, England.

[8] R. Martinez-Bejar, V. Castillo, and F. Martin (1995). Una aproximación basada en conocimiento para el proceso de estudio del paisaje en planificación física de recursos naturales. In R. Rizo y J. M Garcia (Eds.), *Jornadas de Transferencia Tecnológica de Inteligencia Artificial a Industria, Medicina y Aplicaciones Sociales*, TTIA, DTIC, Universidad de Alicante.

[9] MOPT (1992). *Guía para la elaboración de estudios del medio físico.* MOPU, Secretaría de Estado para las Políticas del Agua y el Medio Ambiente, Madrid.

[10] A. M. Starfield, and A. L. Bleloch (1983). Expert Systems: An approach to problems in ecological management that are difficult to quantify. *J. Environ. Mgmt* **16**: 261-268.

[11] R. R. Yager (1984). Approximate reasoning as a basis for rule-based expert systems. *IEEE Transactions on Systems, Man, and Cybernetics* **14**: 636 - 643.

[12] L. A. Zadeh (1983). The role of fuzzy logic in the management of uncertainty in expert systems. *Fuzzy Sets and Systems* **8**(3): 199 - 227.

FREE: Un Entorno Visual para Edición de Consultas Difusas sobre Servidores de Fuzzy SQL*

J. Miguel Medina, David Rashid, J. Luis Torralbo, Rafael Vela. M. Amparo Vila
Dpto. de Ciencias de la Computación e Inteligencia Artificial
E.T.S. de Ingeniería Informática.
Universidad de Granada 18071. Granada (Spain)
email: {medina,vila}@robinson.ugr.es
http://decsai.ugr.es/difuso/difuso.html

Resumen

En este artículo presentamos la arquitectura general de una herramienta visual que permite elaborar y procesar consultas difusas. Esta aplicación opera sobre entorno Windows 3.x o Windows95 y permite acceder a servidores de bases de datos difusos construidos sobre Sistemas de Gestión de Bases de Datos comerciales.

1 Introducción

Desde que Codd introdujera el modelo relacional hemos asistido a una serie de propuestas que persiguen dotar a este modelo de capacidad para representar y manipular información de tipo impreciso. Con esto perseguimos dotar a las bases de datos de mecanismos para albergar un tipo de información más próxima a los esquemas de comunicación y razonamiento humanos. Para plasmar esta capacidad en los modelos de bases de datos, todas las propuestas recurren al empleo de la teoría de conjuntos difusos como herramienta teórica de representación. A partir de aquí aparecen diferentes enfoques sobre como modelar los diferentes aspectos de la información imprecisa y de como integrarlos en el modelo de datos relacional. Existen dos enfoques sobre como representar la imprecisión: 1) utilizar relaciones difusas sobre el dominio de discurso para modelar en que medida resultan próximos dos elementos del mismo respecto a una determinada operación de comparación 2) emplear distribuciones de posibilidad para modelar la imprecisión de los conceptos y utilizar diferentes tipos de medidas para obtener en que grado dos datos imprecisos satisfacen una determinada comparación. La forma de integrar estos dos enfoques para la representación de información imprecisa en el modelo relacional da lugar a diferentes propuestas.

La primera aproximación se encuentra descrita en [1]. Sobre la segunda exsiten varios trabajos, los más importantes fueron publicados por Umano, [12], Prade y Testemale, [11] y Zemankova [13].

En [6, 7], presentamos un modelo, denominado GEFRED, el cual intenta sintetizar los aspectos más relevantes de aproximaciones anteriores en un marco teórico común.

*Este trabajo ha sido financiado por la CICYT con cargo al Proyecto TIC94-1347

El modelo relacional dió lugar a la aparición de diferentes sistemas que trataban de llevar a la práctica sus postulados con el fin de construir, gestionar y mantener bases de datos relacionales. Estos sistemas, denominados genericamente Sistemas Gestores de Bases de Datos Relacionales (SGBDR en lo sucesivo), son de gran complejidad dada la cantidad de tareas que les está encomendadas. La mayoría de ellos proporciona un lenguaje basado en el álgebra y/o cálculo relacional con el que podamos expresar el esquema relacional de una base de datos, introducir y alterar información en la misma y, por supuesto, consultar información. El lenguaje relacional más extendido en la actualidad es SQL.

Para llevar a la práctica un modelo relacional de bases de datos difusas es preciso diseñar un sistema con los mismos cometidos que los SGBDR y que puedan además dar soporte a todas las características propias de la información difusa. Estos sistemas podemos denominarlos genericamente Fuzzy Relational Data Bases Management Systems (SGBDRD en lo sucesivo). En [5, 8] aparecen algunas ideas acerca de como deben ser estos sistemas.

Nosotros pensamos que, tanto el modelo relacional como los SGBDR disponibles en la actualidad proporcionan los recursos necesarios para desarrollar SGBDRD. Estas ideas están planteadas dentro de una arquitectura denominada FIRST, cuya descripción aparece en [8].

Uno de los elementos que dificultan la utilización de las bases de datos difusas es la definición y utilización de conceptos imprecisos en los procesos de consulta. Las interfaces visuales proporcionan al usuario mecanismos más intuitivos e interactivos para expresar sus necesidades de información y para evaluar la bondad de los resultados obtenidos. En este artículo presentamos un módulo de edición y ejecución de consultas difusas utilizando recursos visuales. En el próximo apartado introduciremos este módulo denominado FREE (Fuzzy Request Edition Environment). En la sección 3 describiremos arquitectura de este módulo inscrita dentro del marco de FIRST. En la sección 4 ilustraremos algunos aspectos de su funcionamiento mediante la edición de una consulta, para terminar elaborando las conclusiones derivadas de este

desarrollo y planteando el camino que pretendemos seguir.

2 Una Introducción a FREE

FREE es un entorno gráfico que permite elaborar y resolver consultas difusas sobre un servidor de FSQL. Este programa ha sido desarrollado de acuerdo con la arquitectura Cliente/Servidor. FREE constituye el cliente de este esquema y se ejecuta sobre Windows3.x y Windows95. Las peticiones difusas elaboradas mediante FREE se envían a un Servidor de FSQL (Fuzzy SQL) el cual se encarga de procesarlas y de devolver los posibles resultados para que éste los muestre al usuario. El servidor de FSQL es un módulo que se implementa sobre SGBDR convencionales de acuerdo con las ideas expresadas en [5, 8].

FREE permite la conexión a servidores SQL y FSQL y ofrece mecanismos visuales la elaboración de sentencias SQL y FSQL (en realidad FSQL es un superconjunto de SQL que incluye sentencias para el procesamiento de peticiones difusas). Mediante una serie de pantallas el usuario es conducido desde la selección de las tablas y atributos a utilizar, hasta la elaboración de las condiciones de recuperación. Estas condiciones pueden ser imprecisas o no y pueden ser combinadas conjuntamente. Uno de los aspectos en que se muestra más útil es en la asistencia al usuario en el proceso de decidir que parámetros difusos emplear para obtener información de acuerdo con sus necesidades. Esto se consigue mediante el empleo de ventanas para la visualización y edición gráfica de dichos parámetros.

Mientras vamos construyendo la consulta, FREE va elaborando la sentencia FSQL equivalente que permite expresar esta operación en términos comprensibles por el servidor. Finalizado el proceso de edición visual de la consulta, FREE puede mostrarnos la sentencia FSQL que expresa dicha consulta, esta sentencia puede ser directamente editada por el usuario e incluso almacenada para un uso posterior. A continuación, el usuario puede ordenar la ejecución de la sentencia. Como resultado de esta operación FREE envía la sentencia al servidor FSQL el cual se encarga de procesarla para devolver las tuplas resultantes, que son mostradas al usuario en una ventana sobre la que puede desplazarse para comprobar la información obtenida.

En los procesos de consulta difusa los resultados de una consulta difusa pueden servir como punto de partida de un proceso interactivo en el que el usuario, a la vista de los mismos, altere las condiciones y/o umbrales de cumplimiento con vistas a afinar más los resultados devueltos. Pues bién, este proceso se encuentra adecuadamente soportado por FREE, de tal forma que el usuario puede editar los parámetros de consulta actuales con el fin de ajustarlos para obtener los resultados perseguidos.

3 Implementación de FREE

FREE es un componente de una arquitectura general de SGBDRD denominada FIRST que integra todos los elementos necesarios para la creación, gestión y programación de bases de datos. FIRST es un esquema que reune una serie de ideas para la implementación de SGBDRD utilizando los recursos que proporcionan los SGBDR convencionales. Esta arquitectura utiliza una extensión de SQL denominada Fuzzy SQL (FSQL) como lenguaje de datos del sistema. Este lenguaje integra los sublenguajes de definición de datos (DDL), de manipulacion de datos (DML) y de control de datos (DCL). El acceso al servidor de bases de datos difusas de FIRST solo es posible mediante el empleo de FSQL como lenguaje de datos. El esquema general de esta arquitectura y una sintaxis para FSQL pueden encontrarse en [8]. En [6] pueden encontrarse algunos de los criterios adoptados para la representación e implementación de información difusa utilizando los recursos proporcionados por los SGBDR convencionales.

FREE es un módulo cliente dentro de la arquitectura de FIRST. De acuerdo con esto, FREE se ocupa de asistir la edición de sentencias difusas que, en última instancia, serán expresadas en lenguaje FSQL, directamente procesable por el componente servidor de FIRST.

FIRST está concebido con la idea de que todos sus componentes sean desarrollados utilizando los componentes y recursos que proporciona el SGBDR sobre el que se implementa (este SGBDR se denomina anfitrión). Esos recursos van desde la BD, el servidor de BD, el lenguaje de datos SQL, los lenguajes y herramientas de programación que incorpore (SQL inmerso, API, SQL procedural, etc), los drivers de comunicación sobre red, etc. Todos estos recursos, convenientemente utilizados por una implemtación de FIRST, permiten obtener un SGBDRD con todas las capacidades presentes en el SGBDR anfitrión además de proporcionarle capacidad para el tratamiento información difusa.

El módulo FREE que describimos en este artículo sirve de ejemplo de como explotar las capacidades presentes en un SGBDR concreto, como es Oracle7, para desarrollar un módulo de FIRST completamente integrado en la arquitectura de este SGBDR.

FREE ha sido desarrollado mediante Visual Basic utilizando la tecnología de automatización de OLE 2.0 para implementar el acceso a servidores Oracle7.

3.1 El módulo FREE dentro una Arquitectura FIRST

La figura 1 muestra el esquema de como está integrado FREE dentro de una arquitectura de FIRST construida sobre Oracle7. Este esquema de FIRST incorpora unicamente dos módulos básicos: el servidor de FSQL, implementado sobre el servidor Oracle7 y, el módulo FREE desarrollado sobre Windows. Para dotar a FREE de todas las posibilidades que brinda la arquitectura Cliente/Servidor es preciso el empleo de una serie de componentes. A continuación vamos a describir el cometido de cada uno de los componentes que intervienen en la arquitectura:

FREE Este módulo asiste visualmente al usuario en la elaboración de una consulta (difusa o no). Está desarrollado con Visual Basic y utiliza el soporte para OLE 2.0 de esta herramienta para incorporar acceso a servidores Oracle7.

OLE 2.0 Es el acrónimo de Object Linking & Embbeding un estándar introducido por Microsoft para la comunicación de datos entre aplicaciones

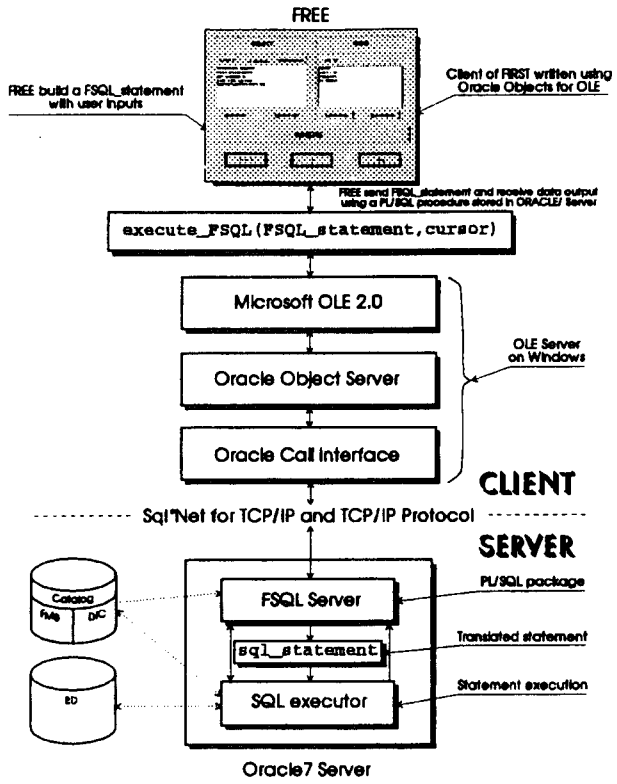

Figura 1: FIRST sobre Oracle7 incorporando el módulo FREE

en entornos Windows. Las aplicaciones pueden soportar OLE en modo cliente, en modo servidor o en ambos. Una aplicación desarrollada con soporte para el modo cliente de OLE puede incorporar en sus documentos información generada con otra aplicación que soporte el modo servidor de OLE. La peculiaridad de OLE consiste en que, cada vez que la aplicación cliente precise editar el contenido de esa información, será invocada de forma automática la aplicación que lo generó. La versión 2 de OLE incorpora la tecnología de automatización. Esta técnica permite que cualquier objeto exponga un conjunto de órdenes y de funciones que puedan ser utilizadas dentro de otra aplicación. Es como si la aplicación servidora proporcione algunas de sus características funcionales a la aplicación cliente.

En el desarrollo de FREE hemos utilizado esta técnica de programación. De un lado, Visual Basic proporciona capacidad para programar aplicaciones que utilicen objetos automatizados y, de otro, las herramientas Oracle Object for OLE proporcionan los objetos que gestionan el acceso al servidor Oracle7. Así pues, incorporamos y utilizamos estos objetos en FREE.

Oracle Object Server Es un servidor OLE 2.0 desarrollado sobre el servidor OLE 2.0 para Windows y resuelve todas las peticiones de acceso a un servidor Oracle7 efectuadas desde una aplicación cliente OLE 2.0.

Oracle Call Interface Los servicios interceptados por el módulo anterior son transformados por él a llamadas soportadas por este interfaz de programación. Este módulo se encarga de establecer la comunicación a través la red mediante el empleo de los drivers nativos de Oracle para el entorno de que se trate.

Sql*Net Éste está construido sobre el protocolo de red e intercepta, de forma transparente, las peticiones dirigidas al servidor de Oracle, liberando al desarrollador de la taréa de implementar, a bajo nivel en las aplicaciones, el acceso a los datos del servidor a través de red.

Servidor de FSQL Se encarga de resolver las peticiones en FSQL y traducirlas a sentencias SQL ejecutables por Oracle. Está desarrollado en PL/SQL e incorpora un procesador sintáctico y semántico para la sintaxis FSQL adoptada. Una vez traducida la sentencia, la envía al ejecutor de SQL del cual obtiene los resultados los cuales, debidamente procesados, son enviados al cliente a través de un mecanismo de comunicación soportado por el concepto de cursor. El servidor de FSQL puede requerir en diversas ocasiones de los servicios del ejecutor de SQL y, por supuesto, de los servicios de la máquina de PL/SQL incorporada en el servidor Oracle (no mostrada en la figura 1). El servidor de FSQL no sólo está escrito en PL/SQL, sino que también se encuentra precompilado y almacenado en el servidor de Oracle.

Es preciso incidir un poco más sobre la técnica empleada para comunicar las sentencias FSQL y los resultados entre el módulo cliente, FREE, y el servidor de FSQL.

En principio, los componentes de OLE2 Server y el driver SQL*Net estan diseñados para permitir un acceso "clásico" al servidor Oracle7 desde una aplicación cliente. Por clásico entendemos que unicamente permiten el envío de sentencias escritas en un lenguaje soportado por el servidor Oracle7. Los únicos lenguajes que soporta este servidor son SQL y PL/SQL. Por tanto, si queremos enviar una sentencia FSQL habremos de encapsularla dentro de estos lenguajes. Afortunadamente, PL/SQL es una extensión procedural de SQL que permite, entre otras muchas otras cosas, la creación e invocación de procedimientos almacenados en la base de datos. Utilizaremos esta característica para construir un paquete PL/SQL que se encargará de dotar al servidor Oracle7 de la capacidad para procesar sentencias FSQL. Este paquete presenta a cualquier cliente FSQL una interfaz común para enviar y procesar sentencias FSQL.

4 Edición y Procesamiento de Consultas Difusas con FREE

Para ilustrar la utilización y funcionamiento de FREE, vamos a detallar como se edita y procesa la siguiente consulta:

"Mostrar el código de empleado, el nombre de empleado, su edad, así como el grado en que se satisface

la condición, de todo el personal "joven" por encima de un umbral 0.8"

Supondremos que tenemos definida la relación *personal* con la información que muestra la tabla 1, donde el atributo *age* permite almacenar información difusa. La figura 2 muestra las distribuciones de posibilidad asociadas a las etiquetas utilizadas sobre este atributo. El valor [25, 30] representa intervalos de edad, el valor [35, 38, 40, 45] es una distribución de posibilidad que indica que el individuo tiene una edad entre 38 y 40 con valor de posibilidad máximo (1), disminuyendo la posibilidad para valores menores de 38 y mayores de 40, siendo ésta 0 para valores menores que 35 o mayores que 45. Por último, [25, 30, 35] y [30, 35, 40] indican que la edad de los indiviudos es de aproximadamente 30 años y aproximadamente 35 años, respectivamente.

EMP#	NAME	SEX	AGE	CITY
E0	VICTOR	Male	BEGINNER	BARCELONA
E1	SIXTO	Male	17	GRANADA
E2	OLGA	Female	27	GRANADA
E3	JOSE	Male	[25,30]	MALAGA
E4	SALVADOR	Male	[35,38,40,45]	VALENCIA
E5	MATILDE	Female	[25,30,35]	MADRID
E6	ANA	Female	UNKNOWN	BARCELONA
E7	CANDIDO	Male	45	GRANADA
E8	LUIS	Male	18	MALAGA
E9	FEDERICO	Male	23	VALENCIA
E10	ALVARO	Male	YOUNG	GRANADA
E11	ELADIO	Male	OLD	MADRID
E12	CARLOS	Male	VERY_OLD	MADRID
E13	CRISTINA	Female	MIDDLE	MALAGA
E14	INES	Female	YOUNG	VALENCIA
E15	JULIAN	Male	31	BARCELONA
E16	ANTONIO	Male	55	MALAGA
E17	SILVIA	Female	[30,35,40]	VALENCIA
E18	PALOMA	Female	27	GRANADA
E19	FERNANDO	Male	YOUNG	BARCELONA
E20	FELIPE	Male	60	VALENCIA
E21	RICARDO	Male	48	VALENCIA
E22	MERCEDES	Female	17	GRANADA

Tabla 1: Tabla *Personal*

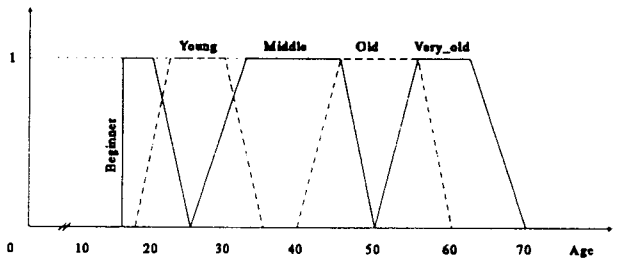

Figura 2: Etiquetas sobre el dominio *Edad*, (en años)

Para editar esta consulta usando FREE los pasos a seguir serán:

1. Invocar al programa FREE, conectarnos con el Servidor Oracle7 deseado e identificarnos como usuario autorizado.
2. Tras esto entraremos en un menú general del que habremos de seleccionar la opción nueva consulta.
3. Aparece la pantalla que muestra la figura 3 donde se muestran las tablas a las que tenemos acceso.
4. Hacemos doble "click" sobre la tabla *personal*. Como resultado aparece una ventana con los campos que contiene esa tabla.

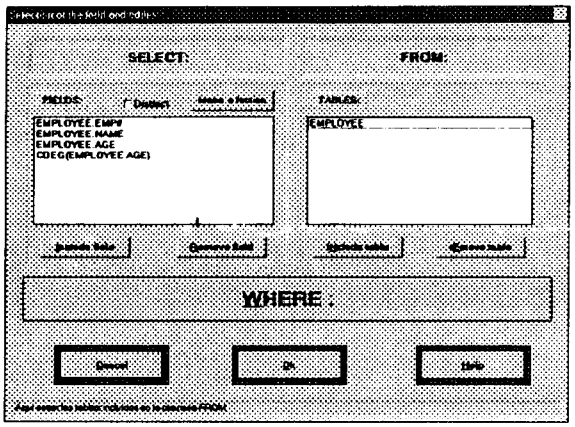

Figura 3: Pantalla de edición de tablas, atributos y expresiones

5. Marcamos los atributos a visualizar: *emp#, nombre, edad.*
6. Para visualizar el grado con que cada tupla satisface la condicion para el campo edad utilizamos el boton **make a function**.
7. La función CDEG indica que se visualice ese grado de cumplimiento. Para ello creamos la expresion CDEG(edad). Aceptamos y observamos que ya tenemos establecida la información a visualizar en la ventana SELECT figura 3.
8. Pinchamos el botón **WHERE** para editar la condición de búsqueda. Aparece la ventana que muestra la figura 4.

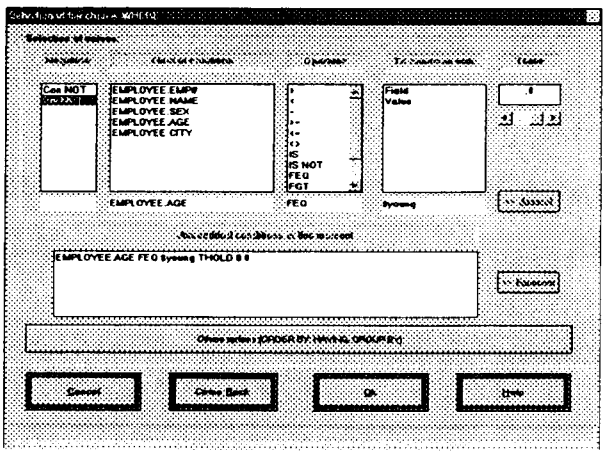

Figura 4: Pantalla de edición condiciones de consulta

9. En esa ventana deben aparecer los atributos de aquellas tablas que hallamos seleccionado en la pantalla anterior, figura 3.
10. Mediante toques de ratón componemos la condición: `age FEQ young`. La etiqueta `young` puede

seleccionarse desde la cuarta columna mostrada en la pantalla. Además es posible editar sus parámetros tal y como muestra la figura 5.

11. Una vez compuesta la condición seleccionamos el umbral 0.8, añadimos y pulsamos el botón **Ok** para aceptar.

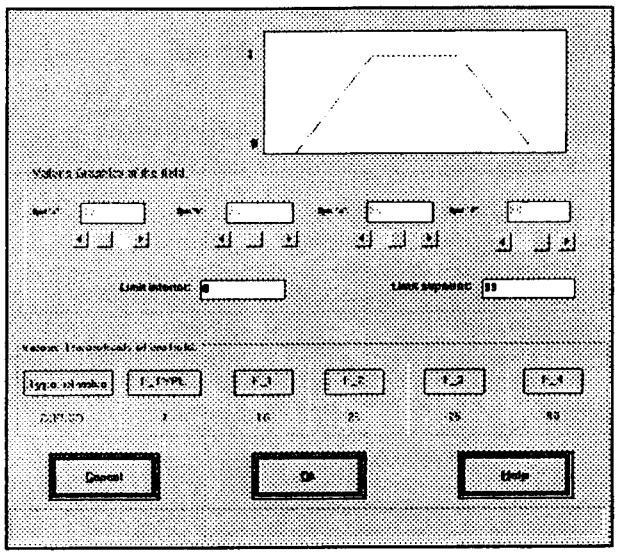

Figura 5: Pantalla de edición de valores difusos

Con esto hemos acabado el proceso de edición de la consulta. Ahora volvemos a la pantalla principal desde la que podemos comprobar que el programa ha generado la siguiente sentencia FSQL:

```
SELECT emp#, name, age, CDEG(age)
FROM employees
WHERE age FEQ $young THOLD 0.8
```

A continuación podemos indicarle a FREE que ejecute la consulta. El resultado obtenido se muestra en la figura 6.

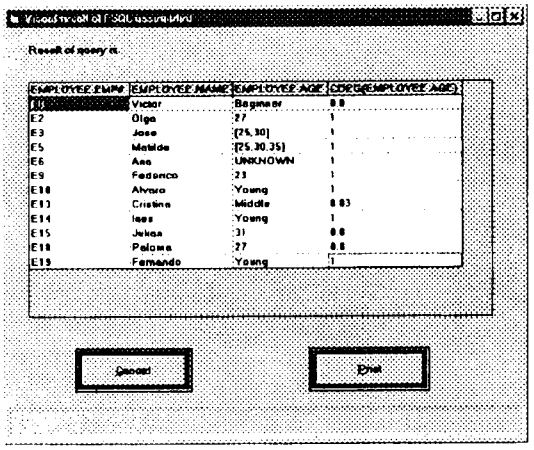

Figura 6: Pantalla con el resultado de la consulta

5 Conclusiones

El programa que hemos presentado en este artículo supone un paso adelante en el camino de integrar el tratamiento de información difusa en los SGBDR. Pone al alcance del usuario final una serie de herramientas para la elaboración visual consultas difusas sobre un servidor de FSQL. Además, está diseñado para integrarse perfectamente dentro de los esquemas de organización de proceso y almacenamiento actuales (arquitecturas Cliente/Servidor, bases de datos distribuidas, etc.). En el desarrollo del módulo FREE y en el de los restantes elementos de la arquitectura en que se integra, se han utilizado los recursos que proporcionan los SGBDR convencionales, así como de herramientas de desarrollo disponibles en la actualidad.

En la actualidad se encuentra en fase de desarrollo un prototipo de servidor de FSQL construido sobre un Servidor Oracle7. Este módulo constituye el elemento central de la arquitectura FIRST y su disponibilidad permitirá avanzar en el desarrollo de herramientas para el desarrollo de aplicaciones que exploten las posibilidades del tratamiento de información difusa en áreas como la medicina, el soporte a la decisión y demás ámbitos donde se requieran patrones flexibles para el tratamiento de la información.

Referencias

[1] Buckles B.P., Petry F.E. "A Fuzzy Representation of Data for Relational Databases". Fuzzy Sets and Systems, 7. 213-226. (1982)

[2] J.C. Cubero and M.A. Vila. "A new definition of fuzzy functional dependencies in fuzzy relational databases". International Journal of Intelligent Systems, 9(5):441–448, (1994).

[3] Hamon Guy. "Extension d'un language d'interrogation de Base de Donnees en vue de l'utilisation de questions imprecises", Tesis Doctoral. (1986).

[4] Li D., Liu. "A Fuzzy Prolog Database System". ed. John Wiley & Sons. New York. (1990)

[5] Medina J. M., Vila M.A., Cubero J.C., Pons O. "Towards the Implementation of a Generalized Fuzzy Relational Database Model". Fuzzy Sets & Systems 75 pp 273-289 (1995)

[6] Medina J. M., Pons O., Vila M.A. "GEFRED. A Generalized Model of Fuzzy Relational Data Bases". Information Sciences,76,1-2, pp 87-109. (1994)

[7] Medina J. M. "Bases de Datos Relacionales Difusas: Modelo Teórico y Aspectos de su Implementación". Tesis Doctoral. Universidad de Granada. http://decsai.ugr.es/difuso.html#tesis (1994)

[8] Medina J. M., Pons O., Vila M.A. "FIRST. A Fuzzy Interface for Relational SysTems". Proceedings of the 6º Sixth International Fuzzy Systems Association World Congress (IFSA 95) vol II pp 409-412. (1995)

[9] Nakajima Hiroshi, Sogoh Taiji, Arao Masaki. "Fuzzy Database Language and Library - Fuzzy Extension to SQL -" Proceedings of the Second IEEE International Conference on Fuzzy Systems, pp. 477-482. (1993)

[10] O. Pons, J.C. Cubero, J.M. Medina, M.A. Vila. "Dealing with Disjunctive Missing Information in Logic Fuzzy Databases" to appear in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems.

[11] Prade H., Testemale C. "Generalizing Database Relational Algebra for the Treatment of Incomplete/Uncertain Information and Vague Queries". Information Sciences, 34. 115-143. (1984)

[12] Umano M. "Freedom−0 : *A* Fuzzy Database System". Fuzzy Information and Decision Processes. Gupta-Sanchez edit. North-Holland Pub. Comp. (1982)

[13] Zemankova M., Kandel A. "Fuzzy Relational Data Bases - A Key to Expert Systems". Verlag TUV Rheinland. (1984)

Optimizador Fuzzy para Paletizado en la Industria Conservera

H.D. Puyosa Piña, P. Soto Alarcón, F. Guardiola Díaz, y L.M. Tomás Balibrea.

Universidad de Murcia, Grupo de Investigación "Visión y Robótica",
Paseo Alfonso XIII, 48, E-30203 Cartagena (Murcia) - España.
hpuyosa@sait01.plc.um.es

Resumen

Este artículo describe el estudio preliminar para la optimización de una operación de paletizado en una industria conservera de la Región de Murcia empleando lógica difusa. El sistema de producción consta de cinco mangas, donde se reciben los botes metálicos clasificados procedentes de una línea de producción, y cinco mesas de paletizado, donde se apilan los botes hasta un total de cinco niveles. Un robot, con capacidad de movimientos en los ejes X e Y, se desplaza hasta las mesas de paletizado y transportando los botes metálicos desde la mesa hasta el pallet mediante sujeción magnética. La secuencia de operaciones se controlada mediante un autómata programable, que dispone de un modulo comercial que implementa lógica fuzzy para resolver los conflictos de atención a las mesas de paletizado.

Palabras Claves: Optimización, Paletizado.

1. Introducción

El empleo de recursos compartidos en las líneas de producción permite abaratar la inversión en maquinarias y equipos, pero exige que el controlador que se instale disponga de mejores prestaciones con el fin de resolver los conflictos que pudieran presentarse en la optimización del recurso. La resolución, en este caso, del conflicto puede realizarse asignando una mayor prioridad a una manga de paletizado sobre las otras, o formando una cola FIFO de atención -de manera que las mangas sean atendidas en el estricto orden en que se van llenando- siendo esta última opción la que se encontraba implementada en planta. Las soluciones anteriormente propuestas son de fácil implementación en un autómata programable, pudiéndose simular y programar utilizando técnicas de Redes de Petri [1]. Sin embargo, estas soluciones ocasionan un bajo rendimiento del manipulador, ya que sus movimientos no están totalmente optimizados, empleando gran parte del tiempo de máquina en desplazamientos que, en casos de elevada producción, pueden derivar en una incapacidad para atender la demanda, produciendo el colapso de la planta.

Nuestra aportación ha consistido en el desarrollo de un algoritmo difuso [2] que permite abordar, con sencillez y robustez, el problema de optimización del mencionado paletizador. Este algoritmo se ha implementado en un modulo fuzzy comercial, que se encarga de decidir, de entre varias mangas llenas, a cual de ellas debe desplazarse el paletizador en función de los siguientes parámetros:

A1 Velocidad de entrada de productos a cada manga.

A2 Tiempo transcurrido desde que se ha llenado cada manga.

A3 Distancia desde la que se encuentra el paletizador a cada una de las mangas que pudieran requerir su atención.

El autómata programable, a través de la permanente comunicación que posee con el módulo fuzzy, transferirá a éste los valores numéricos instantáneos de los parámetros anteriores (variables de entradas) generando el vector de antecedentes del sistema de inferencia borroso. Los parámetros **A1** y **A2** son determinados, respectivamente, a partir de una fotocélula de detección de botes entrantes a la manga y de una fotocélula de saturación de manga. El parámetro **A3** depende de la posición actual del paletizador y de las distancias de separación entre mangas que, en nuestra aplicación, son constantes.

En los apartados siguientes se describe el proceso a optimizar, la formulación del problema, incluyendo las reglas fuzzy, y los resultados obtenidos.

2. Descripción del Proceso

Se pretende automatizar el paletizado de botes de conserva en una planta de fabricación de conservas vegetales en la Región de Murcia. Los botes de conserva se clasifican, en una etapa previa, en función de marcas de color realizadas en su superficie, indicativas del producto y calidad que contienen, mediante un sistema de

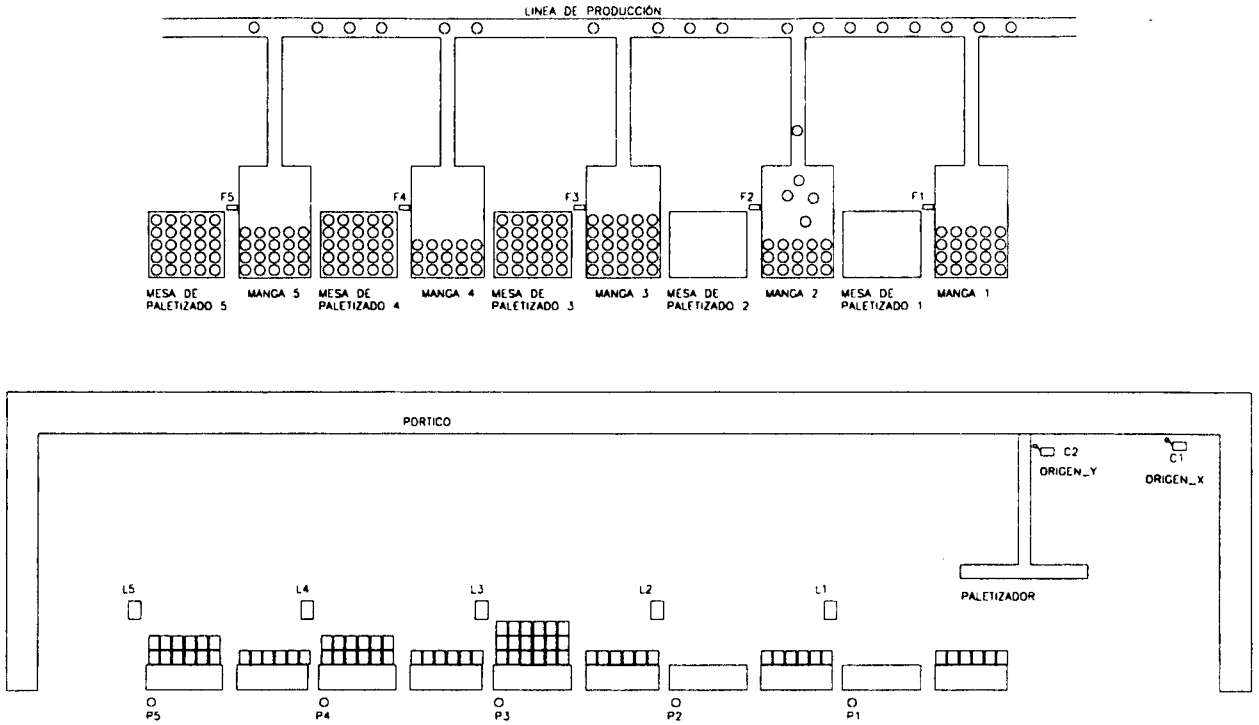

Fig. 1. Diagrama del Proceso

visión por computador. El sistema consta de cinco mangas, donde se reciben los botes clasificados procedentes de una línea de producción, y cinco mesas de paletizado, donde se apilan los botes hasta un total de cinco niveles (Fig. 1). Para realizar la operación se empleará un paletizador de dos ejes (X e Y), los cuales estarán gobernados por dos servo-controladores, uno para cada eje. Al ser los botes de conserva metálicos, la sujeción de éstos para su manipulación se realizará mediante una placa magnética

El control global del paletizador será ejercido por un autómata programable.

Cuando se accione el pulsador de marcha desde el interfaz hombre-máquina, el paletizador realizará en primer lugar una búsqueda de orígenes, determinando su posición, tanto en el eje X como en el eje Y, con la información procedente de los encoders acoplados a los servomotores. Cuando se produzca el llenado de una manga, el paletizador deberá recogerlos y llevarlos a la mesa de paletizado siguiendo la siguiente secuencia:

S1 En primer lugar el paletizador avanzará según el eje X hacia la manga correspondiente.

S2 Una vez que se sitúa sobre la manga desciende, según el eje Y, hasta el nivel en el que se encuentran los botes. Alcanzando éste, se activa la placa de imanes y, transcurrido un segundo desde el inicio de la activación, comienza la elevación.

S3 El paletizador asciende según el eje Y hasta alcanzar el origen (de este eje), comenzando en ese momento el avance hacia la mesa de paletizado donde depositará los botes sobre la mesa en el nivel correspondiente, mediante la desactivación de la placa de imanes, lo que ocurrirá un segundo después de haber alcanzado el nivel correspondiente, transcurrido el cual se iniciará el movimiento de subida en el eje Y.

Para optimizar el trabajo del paletizador se implementa un algoritmo fuzzy sobre un módulo comercial, de manera que resuelva que manga debe ser atendida, teniendo en consideración los valores de los parámetros **A1**, **A2** y **A3** anteriormente descritos. El sistema de control del paletizador se muestra en la Fig. 2.

Las condiciones de operación del proceso de paletizado son las siguientes:

- La velocidad de entrada de los productos a la manga se encuentra comprendida entre 150 botes/minuto y 1300 botes/minuto.
- Desde que se llena una manga, hasta que el paletizador la atiende, no puede transcurrir un tiempo superior a 12 minutos, ya que se produciría el desbordamiento del buffer de la manga.
- La distancia entre mangas es de tres metros.

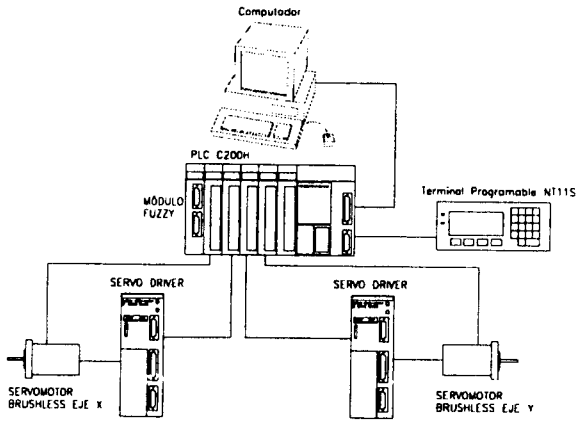

Fig. 2. Sistema de control del paletizador

La salida del modulo fuzzy, una vez desborrosificada, indicará el grado de prioridad de ir a atender a una manga determinada. La manga que tenga una mayor prioridad será la salida atendida. El programa del autómata interroga al modulo fuzzy sobre la conclusión (prioridad) obtenida y en consecuencia actúa para que sea atendida la demanda de una manga llena.

El autómata programable se programa para que cuando la mesa de paletizado se completa con tres niveles de botes, se active un aviso luminoso indicando a un operario que debe proceder a retirar el pallet lleno. El operario, una vez que ha retirado el pallet lleno, procederá a activar un pulsador que causará los efectos de anulación de la señal luminosa, al tiempo que permitirá que el paletizador pueda volver a atender dicha manga.

3. Formulación del Problema

El procedimiento seguido en la realización del optimizador borroso consta de los siguientes pasos:

- Obtención de los datos experimentales para formular las reglas IF-THEN.
- Construcción de la Tabla de la Verdad que representa las conclusiones por implicación de los antecedentes **A1**, **A2** y **A3**.
- Programación del Modulo fuzzy comercial
- Validación de los resultados.

Para cada una de las variables se definieron los siguientes atributos lingüísticos:

Antecedentes
A1 = {VL, VM, VH}
A2 = {TL, TM, TH}
A3 = {DL, DM, DH}

Consecuente
C1 = {PLL, PLM PLH, PMM PMH, PHM, PHH}

En un principio se intentó programar sólo tres etiquetas lingüísticas para el consecuente, de manera que por cada manga existiese una prioridad mínima, una media y otra máxima, ello debido a que resultados previos de varias simulaciones realizadas empleando un software de inferencia para MATLAB [3] apuntaban a que esa partición del universo de salida era suficiente. En la fase de implementación se detectó que las funciones de pertenencia permitidas en el módulo fuzzy comercial eran simplemente líneas verticales (ver [4]-[5]) por lo que la resolución de las soluciones con tres etiquetas no eran aceptables.

Los universos de trabajo para las variables antecedentes se seleccionaron en base a las condiciones de operación del proceso. Así se tiene que:

A1 ∈ [0, 1300] <botes/min>
A2 ∈ [0, 12] < min>
A3 ∈ [1, 15] <mts>

Las funciones de pertenencia para las variables fuzzy del antecedentes se seleccionaron con formas triangulares (Fig.3). En la Tabla 1 se muestran las variables y el código Fismat© empleado en la simulación previa por ordenador.

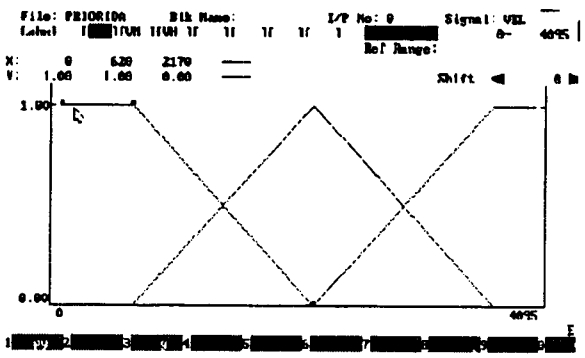

Fig. 3. Función de pertenencia A1

Tabla 1: Función de pertenencia de las variables linguisticas.

Variable	Valor fuzzy	Función de pertenencia
A1	VL =	trapeze(vel, 0.002, 400, 0)
	VM =	trapeze(vel, 0.002, 0, 700)
	VH =	trapeze(vel, 0.002, 200, 1300)
A2	TL =	trapeze(tie, 0.2, 2, 0)
	TM =	trapeze(tie, 0.2, 0, 6)
	TH =	trapeze(tie, 0.2, 2, 12)
A3	DL =	trapeze(dis,0.2, 4, 1)
	DM =	trapeze(dis, 0.2, 0, 8)
	DH =	trapeze(dis,0.2, 4, 15)

En base a la observación del proceso real, que emplea prioridad FIFO de la demanda del paletizador, se construyó la tabla de la verdad o reglas que completan la base del conocimiento del optimizador fuzzy. En la Tabla 2 se presentan las implicaciones que existen entre cada par de funciones antecedentes y la prioridad a serle asignada a cada manga.

Tabla 2a: Reglas de prioridades A1 = VL

A3			
DL	PLL	PLM	PMM
DM	PLL	PLH	PMH
DH	PLM	PMM	PHM

Tabla 2b: Reglas de prioridades A1 = VM

A3			
DL	PLL	PMM	PMH
DM	PLL	PMM	PHM
DH	PLM	PMH	PHH

Tabla 2c: Reglas de prioridades A2 = VH

A3			
DL	PLM	PMM	PHM
DM	PLH	PMH	PHH
DH	PMM	PHM	PHH

En la Fig. 4 se muestra la representación de las reglas bajo el ambiente de desarrollo del modulo fuzzy comercial.

4. Resultados y Conclusiones

Los simulaciones realizadas hasta hoy muestran la potencialidad de este procedimiento en la optimización de paletizado. Se espera que en la implementación real del sistema se obtengan una disminución cercana al 25% de los cuellos de botellas que se están presentando bajo el método de prioridades FIFO.

La aplicación propuesta se enmarca dentro de un proyecto de modernización de la automatización de un paletizador industrial real, que incluye desde la selección del autómata y sus dispositivos de entrada-salida, los servocontroladores de los motores, hasta la programación de la solución propuesta. El abordar el problema de optimización fuzzy del paletizado ha permitido ganar en la profundización de los conceptos de las nuevas técnicas de optimización y control, así como en los problemas que se presentan en la fase de programación, comunicación e intercambio de información entre los distintos componentes que conforman un sistema de control industrial.

Un futuro trabajo a este estudio preliminar abordará la realización de las pruebas en planta. Esa experiencia permitirá, por una parte, establecer índices para la medición del comportamiento del optimizador fuzzy desarrollado, y por otra, ajustar las funciones de pertenencia y las reglas de implicación para aumentar la efectividad del paletizador.

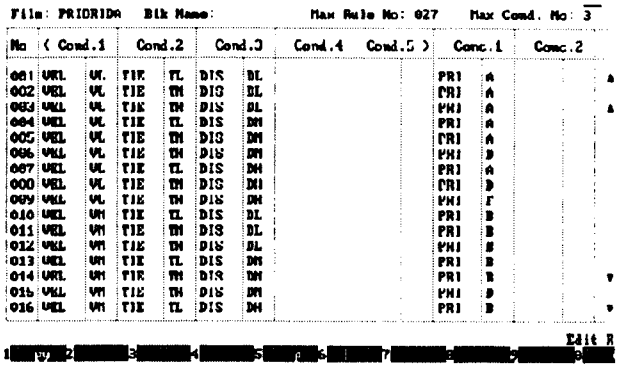

Fig. 4. Reglas de inferencia modulo fuzzy

Referencias

[1] Manuel Silva, "Las Redes de Petri: en la automática y la Informática", Editorial AC, 1985.

[2] Lofti Zadeh, "Outline of a New Approach to the Analysis of Complex Systems and Decision Process", IEEE Trans. Syst. Man and Cybern., vol SMC-3, no. 1, pp. 28-44, 1973.

[3] A. Lofti, "FISMAT: Fuzzy Inference Systems toolbox for MatLab", Universitu of Queensland, 1993. Email: loftia@sl.elec.uq.oz.au

[4] C200H-FZ001 Fuzzy Logic Unit: Operation Manual, OMRON 1992.

[5] Fuzzy Support Software: Operation Manual, OMRON, 1992.