Mecánica Computacional Vol 20. pp. 553—560
Compilado por Fernando G. Flores
Córdoba, 30 de octubre - 2 de noviembre de 2001

UN ALGORITMO GENÉTICO DIFUSO PARA MEJORAR LA DIVERSIDAD GENÉTICA POBLACIONAL

Jessica A. Carballido, Ignacio Ponzoni
Departamento de Ciencias de la Computación
Universidad Nacional del Sur
Av. Alem 1253, 8000, Bahía Blanca, Argentina
e-mail: ip@cs.uns.edu.ar

Diana Brignole
Departamento de Matemática
Universidad Nacional del Sur
Av. Alem 1253, 8000, Bahía Blanca, Argentina

RESUMEN

En este trabajo se presenta un algoritmo genético basado en números triangulares difusos el cual tiene por objetivo superar los problemas de diversidad poblacional observados en los algoritmos genéticos clásicos. En la técnica propuesta se representa cada individuo a través de un conjunto difuso, y se redefinen las operaciones de cruzamiento y mutación para adecuarse a este nuevo patrón genético. Con el fin de establecer el alcance de nuestra propuesta se implementó el algoritmo genético tradicional y el algoritmo genético basado en conjuntos difusos para comparar sus desempeños. Los resultados obtenidos muestran que la variante difusa logra mejor diversidad poblacional sin perder eficacia en la búsqueda de la solución óptima.

ABSTRACT

A genetic algorithm that aims at achieving genetic population diversity in order to avoid the premature convergence of the optimisation problems to undesirable local maxima is presented in this work. The method employs fundamental concepts of diffuse numbers. Each individual is represented by a diffuse number and the crossover and mutation operations are redefined in accordance with the new genetic pattern. The scope of the new proposal was assessed by comparing its performance with the corresponding traditional genetic algorithm. The results show that the variant based on diffuse sets succeeds in achieving better population diversity without losing much selective pressure in the search for the optimal solution.

INTRODUCCIÓN

Uno de las disciplinas dentro de las ciencias de la computación con más fuerte expansión en las últimas décadas a sido la computación evolutiva (CE) [1], [2][3] y [4]). Esta comprende distintos enfoques de simulación de la evolución: ***algoritmos genéticos (AG)***, estrategias evolutivas y programación evolutiva. Todos ellos tienen en común la reproducción (cruzamiento), modificaciones aleatorias (mutación), competencia y selección de individuos que rivalizan en una población.

En particular, los AG son métodos de búsqueda que utilizan operadores basados en la genética natural para explorar el espacio de búsqueda. En términos generales, los AG resultan robustos, eficientes y eficaces en problemas de optimización complejos, aunque también poseen otros áreas de aplicación tales como tareas de planificación (routing, scheduling, packing, etc), diseño, simulación e identificación, control y clasificación.

Los AGs procesan una población de individuos que representan potenciales soluciones del problema tratado. Esta población evoluciona con cada nueva generación hasta alcanzar una solución. Este proceso evolutivo emplea tres operadores básicos: ***selección***, ***cruzamiento*** y ***mutación***. El operador de selección favorece la supervivencia a aquellos individuos con "mejores" características. Por otro lado, los operadores de mutación y cruzamiento crean nuevos individuos, conformados con parte del código genético de sus padres, a fin de recorrer regiones aún no exploradas.

La ***diversidad poblacional*** y la ***presión selectiva*** constituyen dos de los factores más importantes en el proceso de búsqueda genética. Por un lado, la diversidad poblacional refiere a la variedad de esquemas genéticos presentes en la población. A mayor cantidad de esquemas, mayor es la diversidad poblacional. En cambio, la presión selectiva es consecuencia directa de las características del operador de selección empleado. Cuando más se favorezca la supervivencia de los individuos con mejor fitness mayor presión selectiva se estará ejerciendo. Ambos conceptos están estrechamente vinculados ya que un aumento de la presión selectiva ocasiona una disminución de la diversidad poblacional. Esto se debe a que los individuos con mejor fitness son preferidos para el cruzamiento, provocando que la población tienda a converger a los esquemas de cromosomas derivados de estos individuos. Inversamente, un aumento en la diversidad poblacional conlleva a una reducción de la presión selectiva.

Estos aspectos poseen una fuerte incidencia en los resultados obtenidos por el AG. Por un lado, si se está en presencia de un AG con una fuerte presión selectiva se corre el riesgo de alcanzar una convergencia prematura a un máximo (o mínimo) local sin encontrar el máximo (mínimo) global. Esto se debe a la pérdida de diversidad poblacional. Muchas veces sucede que un individuo con bajo fitness posee uno o más genes que forman parte del código genético de la solución óptima. Sin embargo, este individuo desaparece sin dejar descendientes dado que una fuerte presión selectiva sólo favorece el cruzamiento de los mejores miembros de la población, y esto provoca que su esquema genético se pierda para siempre. Por otra parte, si el AG preserva una gran diversidad poblacional, la convergencia puede tornarse muy lenta debido al aumento del espacio de búsqueda explorado. Luego, la mejor alternativa parece ser una posición intermedia, con niveles de presión selectiva y diversidad poblacional equilibrados. Sin embargo, cada problema tiene características propias, las cuales hacen difícil establecer en general un nivel de balance apropiado para estos factores. Esto último constituye la principal motivación del presente trabajo, en donde se propone una variante difusa del AG tradicional con el fin de lograr una metodología que posea una buena diversidad poblacional sin perder demasiada presión selectiva, y así evitar la convergencia prematura.

ALGORITMOS GENÉTICOS Y CONJUNTOS DIFUSOS

La lógica y teoría de subconjuntos difusos fue introducida por L.Zadeh [5] como una herramienta para tratar con problemas caracterizados por la presencia de imprecisión, inexactitud, o incertidumbre. En la matemática clásica, dado un elemento x y un subconjunto X de un cierto universo U, ó x pertenece a X, ó x no pertenece a X. Zadeh admite el grado de prtenencia de un elemento a un subconjunto borroso X del universo U, aceptando así la imprecisión.

Un subconjunto borroso $\overline{X}$ de U, queda definido por su función de pertenencia:

f $\overline{X}$ definida en U y tomando como valores los números reales entre 0 y 1, siendo
f $\overline{X}$ (x) el grado de pertenencia del elemento x a $\overline{X}$.

Un tipo especial de subconjuntos difusos del conjunto **R** de los números reales, son los números difusos triangulares que se indican por la notación:

$\bar{X} = [x_1, x_2, x_3]$, x_1, x_2, x_3 números reales, $x_1 < x_2 < x_3$

siendo su función de pertenencia:

$$f_{\bar{X}}(x) = \begin{cases} \dfrac{x - x_1}{x_2 - x_1} & si \quad x_1 < x \leq x_2 \\ \dfrac{x - x_3}{x_2 - x_3} & si \quad x_2 < x \leq x_3 \\ 0 & si \quad x < x_1 \, ó \, x > x_3 \end{cases}$$

es decir x_2 posee el mayor grado de pertenencia a $\bar{X}$, y este grado disminuye a medida que x se aleja de x_2, acercándose a los extremos del intervalo $[x_1, x_3]$, donde será nulo.

En todo algoritmo o proceso en que se introduzcan variables fuzzy, se obtendrá como resultado un subconjunto fuzzy, y deberá recurrirse a algún procedimiento de "defuzzyficación". En Klir- BoYuan [6], pueden encontrarse los métodos más usuales. En general, la bondad del procedimiento elegido dependerá de la aplicación en particular.

En diversos trabajos se han propuesto diferentes técnicas que integran los conceptos de AG y Matemática Difusa [7], [8], [9], [10] y [11]. Dentro de las distintas técnicas propuestas se pueden diferenciar dos enfoques. Por un lado, el empleo de los AG como método para la optimización de las reglas de inferencia difusas utilizadas por los controladores borrosos. Aquí, los AG son utilizados para optimizar las funciones de pertenencia empleadas por los modelos difusos. Dentro de esta rama se enmarcan los trabajos de Bastian [9], que utiliza programación genética para identificar modelos matemáticos difusos, Wu y Yu [10], en donde se propone un AG basado en aprendizaje para la identificación de modelos TSK, y Huang y Wang [11], que desarrollaron un algoritmo genético de macroevolución adaptiva para diseños de modelos borrosos.

La segunda rama, aborda la integración en sentido inverso, presentando diferentes alternativas para la fuzzyficación o introducción de los conceptos de la matemática difusa en los AG clásicos. De este modo, han surgido diversas variantes difusas de los AG. Ejemplos de estos desarrollos son las investigaciones de Buckley y Hayashi [7], quienes presentan un Algoritmo Genético Difuso (AGD) con una representación fuzzificada para los genes de los individuos, y Herrera *et al.* [8], en donde se proponen operadores genéticos difusos para una clase particular de AG conocidos como Real-Code Genetic Algorithms (RCGA).

Nuestra propuesta se enmarca dentro de esta segunda línea de investigación, presentándose un AGD cuyo principal objetivo es superar los problemas de diversidad poblacional observados en los AG tradicionales. El nuevo algoritmo emplea una representación de los cromosomas diferente a la planteada por Buckley y Hayashi [7], en donde cada individuo es ahora caracterizado a través de un número difuso. Asimismo, los operadores genéticos tradicionales son redefinidos en términos de la nueva estructura genética.

ALORITMOS GENÉTICOS TRADICIONALES

El los AG clásicos, un individuo de la población es representado mediante su cromosoma, el cual es usualmente codificado como una cadena binaria de unos y ceros. Cada bit de esta secuencia constituye un gene del patrón genético. El cromosoma también se denomina ***genotipo*** del individuo,

mientras que el valor asociado a esta estructura constituye su ***fenotipo***. Asimismo, cada individuo posee un "fitness", el cual constituye una medida de las bondades del individuo como posible solución al problema. En términos generales, sea t el número de generaciones y P(t) la población en la t-ésima generación, el AG puede expresarse como sigue:

$t \leftarrow 0$; {*t representa el número de generaciones*}
Inicializar la población inicial P(t);
Calcular el fitness de los individuos de P(t);
Mientras la condición de terminación no se satisfaga hacer:
  $t \leftarrow t+1$;
  Seleccionar una subpoblación S(t) de individuos de P(t-1);
  Efectuar el cruzamiento y mutación de los miembros de S(t) formando una subpoblación S'(t);
  Calcular el fitness de los individuos de S'(t);
  Obtener P(t) sustituyendo algunos individuos de P(t-1) por los individuos de S'(t);
fin-mientras

Como se puede observar, en cada iteración t del AG se selecciona un subconjunto S de individuos de P(t) los cuales son cruzados y/o mutados. Esta selección se realiza de forma tal que aquellos cromosomas con mejor fitness poseen mayores posibilidades de integrar S. Posteriormente, estos nuevos individuos son incorporados a la generación $t+1$ de la población. Finalmente, y con el objeto de mantener estable la población, se elimina un número igual de individuos de la t-ésima generación. Estos pasos se repiten hasta satisfacer alguna condición de terminación predeterminada.

PLANTEO DIFUSO DEL PROBLEMA DE OPTIMIZACIÓN

El primer paso hacia la fuzzyficación de los AG es establecer un planteo para el problema de optimizar conjuntos difusos. En particular, para problemas de maximización se puede establecer la siguiente formulación:

Sea F tal que: $\overline{Y} = F(\overline{X})$
donde $\overline{X}$ es un subconjunto difuso para algún intervalo $[0, M] \subseteq \mathbf{R}$ con $M > 0$.
$\overline{Y}$ es un subconjunto difuso de **R**.

Nuestro problema consiste en encontrar el $\overline{X}$ en [0, M] tal que "maximiza" $\overline{Y}$. Sin embargo, no es posible maximizar directamente $\overline{Y}$ dado que se trata de un conjunto difuso. Por tal motivo, resulta necesario definir una función M que permita "medir" cuan grande es un determinado $\overline{Y}$. Sea $M(\overline{Y}) = \Theta$, donde Θ es un subconjunto de **R**. De este modo, el problema ahora consiste en encontrar un $\overline{X}$ que maximice Θ, quedando nuestro problema de optimización replanteado como:

$$F(\overline{X}) = \Theta$$

FUZZYFICACIÓN DEL ALGORITMO GENÉTICO CLÁSICO

A fin de obtener una fuzzificación de los AG es necesario establecer en primer lugar la representación binaria de un número difuso. Esto constituye la base para la representación del cromosoma de un individuo.

REPRESENTACIÓN BINARIA DE UN CROMOSOMA DIFUSO

Sea $\overline{X}$ un número difuso trianular, $\overline{X} = [x_1,x_2,x_3]$, diremos que su correspondiente representación binaria difusa es $\overline{B} = [b_1,b_2,b_3]$ donde b_1, b_2 y b_3 son los genotipos de los fenotipos x_1, x_2 y x_3 respectivamente.

Asimismo, el fitness de un cromosoma difuso $\overline{X}$ es definido como el $M(F(\overline{X}))$. Dada está definición, el operador de selección para los algoritmos genéticos difusos (AGD) resulta idéntico al de los AG clásicos dado que emplea el valor real correspondiente a $M(F(\overline{X}))$. Sin embargo, la redefinición de la estructura de un cromosoma como una terna de números binarios torna necesario un replanteo de los operadores de cruzamiento y mutación.

Operador de Cruzamiento

Sean $\overline{X}_1$ y $\overline{X}_2$ dos individuos cuyos cromosomas son $\overline{B}_1 = [b_{11},b_{12},b_{13}]$ y $\overline{B}_2 = [b_{21},b_{22},b_{23}]$, y sea n la cantidad de bits empleados para la representación binaria. Un cruzamiento de un punto generará dos nuevos individuo $\overline{X}_3$ y $\overline{X}_4$ cuyos cromosomas $\overline{B}_3 = [b_{31},b_{32},b_{33}]$ y $\overline{B}_4 = [b_{41},b_{42},b_{43}]$ serán tales que los primeros n/2 genes de b_{31}, b_{32} y b_{33} serán iguales a los n/2 primeros genes de b_{11}, b_{12} y b_{13} y los n/2 últimos genes de b_{31}, b_{32} y b_{33} serán iguales a los n/2 últimos genes de b_{21}, b_{22} y b_{23}. Mientras que los primeros n/2 genes de b_{41}, b_{42} y b_{43} serán iguales a los n/2 primeros genes de b_{21}, b_{22} y b_{23} y los n/2 últimos genes de b_{41}, b_{42} y b_{43} serán iguales a los n/2 últimos genes de b_{11}, b_{12} y b_{13}. El cruzamiento de cromosomas difusos puede generar un hijo $\overline{B} = [b_1,b_2,b_3]$ en donde no se verifique que $b_1 \leq b_2 \leq b_3$. En tales casos, resulta necesario reordenar los b_i para obtener un número triangular difuso consistente.

La última etapa del cruzamiento consiste en analizar el fitness de cada b_i y construir su cromosoma difuso de la siguiente forma:

1) Si (fitness(b_1) y fitness(b_2)) ≤ fitness(b_3) entonces el $\overline{B} = [b_1,b_3,b_3]$
2) Si (fitness(b_3) y fitness(b_2)) ≤ fitness(b_1) entonces el $\overline{B} = [b_1,b_1,b_3]$
3) Si (fitness(b_1) y fitness(b_3)) ≤ fitness(b_2) entonces el $\overline{B} = [b_1,b_2,b_3]$

Esta regla asegura que el centro del número difuso corresponda al b_i que posee mejor fitness. De este modo, no se modifica el rango de valores correspondiente al conjunto difuso, lo cual preserva los niveles de diversidad poblacional, y se aumenta la presión selectiva.

Operador de Mutación

Sea A un individuo cuyo cromosoma es [b_1, b_2, b_3]. La mutación genética de A puede producirse en cualquiera de los cromosomas de la terna, e incluso en varios de ellos a la vez. Básicamente, la mutación para números triangulares difusos aplica el operador de mutación tradicional a cada uno de los b_i. Nuevamente, si luego de la mutación, no se preserva la relación $b_1 \leq b_2 \leq b_3$, un reordenamiento de los b_i resultará necesario.

ALGORITMO GENÉTICO BASADO EN NÚMEROS DIFUSOS

Habiéndose definido los conceptos de cromosoma, cruzamiento y mutación basados en números triangulares difusos se presenta a continuación el AGD:

```
t ← 0;  {t representa el número de generaciones}
Inicializar la población difusa inicial P(t);
Calcular el fitness de los individuos de P(t);
Mientras la condición de terminación no se satisfaga hacer:
  t ← t+1;
  Seleccionar una subpoblación difusa S(t) de individuos de P(t-1);
  Efectur el cruzamiento y mutación difusa de los miembros de S( t)
    formando una subpoblación difusa S'(t);
  Calcular el fitness de los individuos de S'(t);
  Obtener P(t) sustituyendo algunos individuos de P(t-1)
    por los individuos de S'(t);
fin-mientras
```

Como puede apreciarse, el AGD tiene la estructura del AG tradicional, sólo que la representación de los cromosomas y los operadores genéticos ahora son difusos.

ANÁLISIS COMPARATIVO ENTRE EL AG TRADICIONAL Y EL AGD

Para analizar el desempeño del AGD frente al AG tradicional (ó AG), se implementaron ambos algoritmos en lenguaje C, y se utilizaron ambas técnicas para resolver varios casos de estudio. En todos los casos, el problema consistió en obtener el óptimo de una función.

La función empleada para la desfuzzyficación de un individuo difuso (ID) fue la siguiente:

Valor Desfuzzycado = (ID.fmay+3*ID.fmed+ID.fmen) / 4

donde ID.fmen es el fenotipo correspondiente cromosoma menor de ID,
ID.fmed es el fenotipo correspondiente al cromosoma medio de ID,
ID.fmay es el fenotipo correspondiente al cromosoma mayor de ID,

Es importante aclarar que se definió esta forma de desfuzzificación para ponderar con mayor peso al fenotipo del centro. Esta elección se debe a que, como se explicó anteriormente, el cruzamiento difuso asegura que el valor del medio poseerá siempre un fitness mayor o igual al de los extremos del conjunto borroso.

DESCRIPCIÓN DE LOS CASOS DE ESTUDIO

Ambos algoritmos fueron testeados con una amplia variedad de ejemplos. En general se observó que las principales diferencias de desempeño se presentan para funciones discontinuas, cuyos máximos locales dificultan la convergencia del AG hacia el máximo global. Por esta razón , los resultados aquí reportados se basan en la optimización de las siguientes funciones f: $R \rightarrow R$:

Caso de Estudio 1:

$$f(x) = x*\text{sen}(10*\pi*x)+1{,}0 \qquad \text{donde } x \in [-1..2]$$

Caso de Estudio 2:

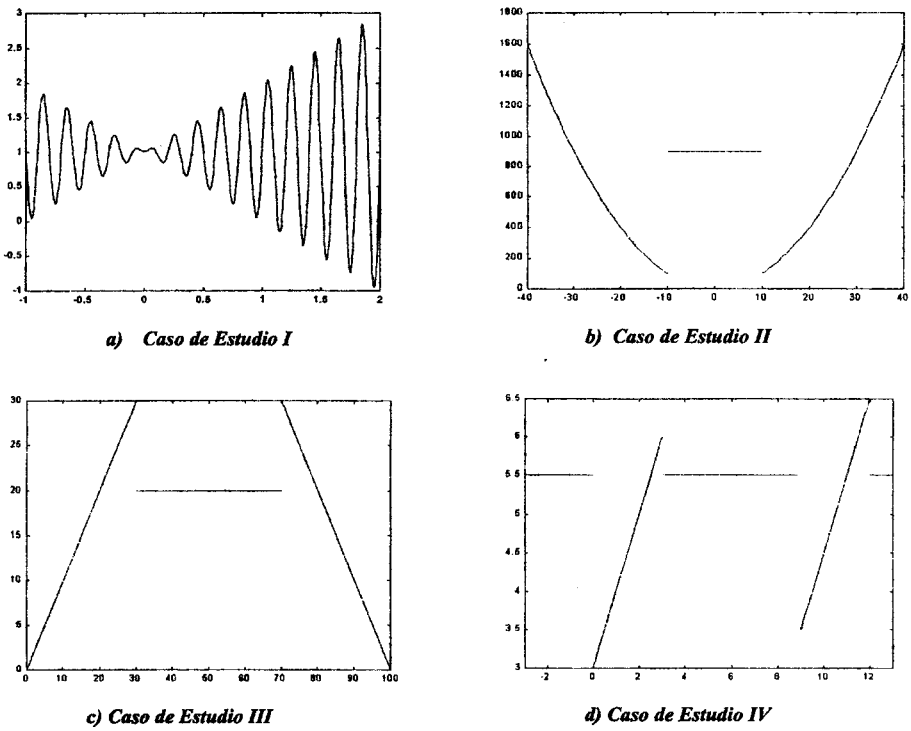
$$f(x) = \begin{cases} x \text{ si } x \le 30 \\ 20 \text{ si } 30 < x \le 70 \\ 100 - x \text{ si } x > 70 \end{cases} \qquad \text{donde } x \in [0..100]$$

Caso de Estudio 3:

$$f(x) = \begin{cases} x^2 \text{ si } x \le -10 \\ 900 \text{ si } -10 < x < 10 \\ x^2 + x*0{,}1 \text{ si } x \ge 10 \end{cases} \qquad \text{donde } x \in [-40..40]$$

Caso de Estudio 4:

$$f(x) = \begin{cases} 5{,}5 \text{ si } x < 0 \\ x+3 \text{ si } 0 \le x < 3 \\ 5{,}5 \text{ si } 3 \le x < 9 \\ x-5{,}5 \text{ si } 9 \le x \le 12 \\ 5{,}5 \text{ si } x > 12 \end{cases} \qquad \text{donde } x \in [-3..13]$$



a) Caso de Estudio I

b) Caso de Estudio II

c) Caso de Estudio III

d) Caso de Estudio IV

Figura 1. *Gráficos de las funciones correspondientes a los casos de estudio.*

RESULTADOS

Los cuatro casos de estudio fueron resueltos por ambos algoritmos empleando probabilidades de 0,15 y 0,8 para la mutación y el cruzamiento respectivamente. La tabla I muestra los resultados obtenidos. Las primeras tres columnas identifican el caso de estudio, la cantidad de generaciones y tamaño de la población utilizados en cada corrida. Las columnas centrales muestran los mejores individuos y sus respectivos fitness hallados por ambos algoritmos. Finalmente, las últimas dos columnas muestran los valores de x y f(x) correspondientes al máximo global de la función.

Tabla I. Resultados

Función	Cant. Gen.	Tam. Pob.	AGD		AG		x	f(x)
			Valor	fitness	Valor	fitness		
CE I	200	20	1,850	2,850	1,850	2,850	1,850	2,850
CE II	100	30	30,000	30,000	70,000	29,990	30,000	30,000
CE II	200	30	30,000	30,000	70,000	29,990	30,000	30,000
CE III	100	20	40,000	1604,000	-39,990	1599,207	40,000	1604,000
CE III	200	20	40,000	1604,000	-40,000	1599,999	40,000	1604,000
CE IV	100	50	12,000	6,500	12,000	6,500	12,000	6,500

En los casos de estudio 1 y 4 se puede observar que ambos algoritmos obtienen el óptimo de la función. Para los ejemplos 2 y 3, el AGD alcanzó el máximo global mientras que el AG convergió a un máximo local. Observando los gráficos de estas dos funciones, se aprecia que el peor desempeño del AG se debe a que su fuerte presión selectiva concentra la búsqueda en el sector del dominio

correspondiente al máximo local, lo cual conduce hacia una convergencia prematura. Esta situación se mantiene aún luego de duplicar la cantidad de generaciones a explorar.

CONCLUSIONES

En este trabajo se presentó una propuesta de fuzzificación de los algoritmos genéticos cuyo principal objetivo fue superar los problemas de convergencia prematura exhibidos en los algoritmos genéticos tradicionales. Nuestro enfoque apunta a mejorar la diversidad poblacional mediante una nueva representación de los individuos de la población basada en el uso de conjuntos difusos. Junto con esta variante en la estructura genética de los cromosomas, se realizó una redefinición de los operadores genéticos de cruzamiento y mutación.

El algoritmo genético basado en números triangulares difusos fue implementado y comparado con el tradicional. Los resultados obtenidos para varios casos de estudio revelaron que la nueva propuesta posee efectivamente una mejor diversidad genética, logrando así superar los problemas de convergencia prematura presentes en estos ejemplos.

A futuro se proyecta experimentar con distintas estrategias de desfuzzificación y proponer operadores genéticos difusos alternativos que permitan acentuar los logros alcanzados por este nuevo método. En particular, se está comenzando a experimentar con operadores de cruzamiento que intersectan las zonas de mayor fitness dentro de los números triangulares difusos.

REFERENCIAS

[1] **Goldberg, D.E.** *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Pub Co, 1989.

[2] **Michalewicz, Z.** *Genetic Algorithms + Data Structures = Evolution Programs*, 3ra edición revisada, Springer Verlag, 1996.

[3] **Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.** *Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications*, Morgan Kaufmann Publishers, 1998.

[4] **Gen, M., Cheng, R.** *Genetic Algorithms and Engineering Optimization*, Wiley Series in Engineering Design and Automation, John Wiley & Sons, 2000.

[5] **Zadeh, L.A.,** *Fuzzy Sets*, Inf. Control 8, 1965.

[6] **Klir, G. & Yuan, B.** *Fuzzy Sets and Fuzzy Logic, Theory and Applications*, Prentice Hall, 1995.

[7] **Buckley, J.J., Hayashi, Y.,** *Fuzzy Genetic Algorithm and Applications*, Fuzzy Sets and Systems, 61, 129-136, 1994.

[8] **Herrera, F., Lozano, M., Verdegay, J.L.** *Fuzzy Connectives Based Crossover Operators to Model Genetic Algorithms Population Diversity*, 92, 1997, págs.21-30.

[9] **Bastian, A.,** *Identifying Fuzzy Models Utilizing Genetic Programming*, Fuzzy Sets and Systems, 113, 2000, págs. 333-350.

[10] **Wu, B., Yu, X.,** *Fuzzy Modelling and Identification with Genetic Algorithm Based Learning*, Fuzzy Sets and Systems, 113, 2000, págs. 351-365.

[11] **Huang, Y.-P., Wang, S.-F.,** *Designing a Fuzzy Model by Adaptive Macroevaluation Genetic Algorithms*, Fuzzy Sets and Systems, 113, 2000, págs. 367-379.