*ÁGORA — Papeles de Filosofía —* (2005), 24/2: 29-48 ISSN 0211-6642

# DE LOS CONJUNTOS FUZZY A LA SOFT COMPUTING[1]

José Luis Verdegay
*Universidad de Granada*

**Resumen**
En este artículo se consideran, analizan y revisan las principales componentes de la "Soft Computing" a partir de la definición original de L.A. Zadeh. Entonces se justifica un nuevo conjunto de integrantes de la Soft Computing en el segundo nivel, en el que destaca la incorporación de las Metaheurísticas en lugar de los Algoritmos Genéticos.
*Palabras clave*: Soft Computing, fuzzy sets, metaheurísticas

**Abstract**
The main constituents of Soft Computing as defined by L.A. Zadeh are considered, analyzed and revised. Then a new set of basic ingredients at the second level is justified and proposed. Among these new components, the presence of Metaheuristics instead of Genetic Algorithms is to be pointed out.
*Keywords*: Soft Computing, fuzzy sets, metaheurísticas

> "Los conceptos de unidad y conjunto son conceptos primarios al espíritu humano: no se pueden descomponer en otros mas sencillos. No se pueden, por tanto, definir." (Constantino Marcos y Jacinto Martínez: Matemáticas, Quinto Curso. Ediciones S.M., 1967)

## 1. Introducción

Desde que en 1965 Lotfi A. Zadeh introdujera el concepto de conjunto *fuzzy* permitiendo la pertenencia de un elemento a un conjunto de forma

*Recibido: 06/09/06. Aceptado:26/10/06.*

[1] Trabajo financiado por los proyectos TIC2002-10886-E (HEUR), TIN2005-24790-E (RESCATE), TIC-00129-JA (MINAS) y TIN2005-08404-C04-01 (HeuriFuzzy),

gradual, y no de manera absoluta como establece la teoría conjuntista clásica, es decir, admitiendo pertenencias valoradas en el intervalo [0,1] en lugar de en el conjunto {0,1}, las aplicaciones y desarrollos basados en este sencillo concepto han evolucionado de tal modo que, hoy en día, es prácticamente imposible calcular el volumen de negocio que generan en todo el mundo, pudiendo encontrar productos cuyo funcionamiento está directamente basado en dicho concepto desde los más usuales electrodomésticos, lavadoras, microondas, cámaras fotográficas, ..., hasta los más sofisticados sistemas, frenado de trenes, control de hornos, navegación automática, ...

La necesidad de encontrar la solución optimal de un problema correctamente planteado, o la mejor solución entre las disponibles, justifica que se construyan y estudien teorías, y se propongan metodologías adecuadas al campo científico en el que surge la cuestión que se ha de resolver. Desde un punto de vista más concreto, pero aún muy general, una importante clase de problemas son los conocidos con el nombre de problemas de optimización, habitualmente asociados a tener que encontrar el máximo o el mínimo valor que una determinada función puede alcanzar en un cierto conjunto previamente especificado. Todo lo relativo a estos problemas se enmarca dentro del cuerpo doctrinal denominado Programación Matemática, que incluye una enorme variedad de situaciones, según que se consideren casos lineales, no lineales, aleatoriedad, un solo decisor o varios decisores, etc. Entre todos los modelos que se incluyen en la Programación Matemática, el más y mejor estudiado, así como el que ha probado tener unas repercusiones prácticas más importantes, es el correspondiente al caso lineal uni-objetivo, tema del que se ocupa la Programación Lineal. Los métodos y modelos de la Programación Lineal tienen relevantes aplicaciones en las diferentes áreas de las Ingenierías, la Economía, las Matemáticas, la Investigación Operativa, la Inteligencia Artificial, y demás disciplinas más o menos relacionadas con la optimización, y constituyen un sustrato teórico más que adecuado para abordar de un modo elegante y eficiente situaciones muy complejas.

Cuando en los problemas de Programación Matemática se consideran elementos de naturaleza borrosa, surgen los métodos de optimización borrosa, quizás una de las áreas mas fructíferas en el ámbito de lo *fuzzy*, tanto desde el punto de vista teórico, como aplicado, que aunque recoge métodos y modelos que dan solución a una enorme variedad de situaciones prácticas reales, del mismo modo que le ocurre a la Programación Matemática convencional, no puede dar respuestas en todos los escenarios posibles. Y no puede hacerlo por una sencilla razón, y es que

hay problemas que siendo planteables en términos propios de ese campo, no son resolubles con sus técnicas.

La facilidad de resolver problemas reales de dimensión cada vez mayor, gracias a la mayor potencia y el menor costo de los computadores, la imposibilidad de conocer en todos los casos las soluciones exactas que les corresponden a esos problemas, y la necesidad de dar respuestas a las situaciones prácticas contempladas en multitud de casos (problemas de organización de las tareas que ha de efectuar un robot, de identificación de itinerarios, de clasificación y ubicación de recursos, de recorte, … en definitiva, problemas combinatorios), han motivado que los algoritmos de tipo heurístico sean empleados cada vez más, como valiosas herramientas capaces de proporcionar soluciones donde los algoritmos exactos no son capaces de encontrarlas. Así en los últimos años ha surgido un largo catálogo de técnicas diversas, animadas por el principio de que es mejor satisfacer que optimizar, o lo que es lo mismo que, antes que no poder dar la solución optima a un problema, es mejor dar una solución que satisfaga al usuario en algún sentido que previamente habrá especificado, y que se han demostrado extraordinariamente efectivas. Ejemplos de esas técnicas pueden ser los algoritmos de Búsqueda Tabú, Enfriamiento Simulado, GRASP ("Greedy Randomized Adaptive Search Procedure"), Genéticos, o los mas recientes: Meméticos, VNS (Búsqueda por Entornos Variables), Colonias de Hormigas, Estimación de Distribuciones, Búsqueda Dispersa, Programación por Restricciones, que en definitiva, demuestran el gran interés de este campo, y la falta de un marco teórico en el que encuadrar, relacionar y poder comparar estos algoritmos.

Se puede decir que en la mayoría de los casos, estas heurísticas se han inspirado en algún modelo real de la naturaleza, la sociedad, la física, ... para producir modelos teóricos que se ajustan a las circunstancias consideradas. Desde esta perspectiva se ha conseguido proporcionar solución a casos que, hace muy poco tiempo, eran intratables con las técnicas convencionales. Sin embargo, las soluciones conseguidas no han sido en la inmensa mayoría de los casos las óptimas. Han sido soluciones "cercanas a las óptimas", que frecuentemente se han obtenido con cargo a criterios distintos del clásico "conseguir el mejor valor de la función objetivo", al considerar características subjetivamente establecidas por el decisor.

Como es ya de sobra reconocido, cuando hablamos de subjetividad asociada a personas humanas, o incluso de cercanía a un valor óptimo, la forma mejor contrastada de modelar ese tipo de situaciones es mediante los conjuntos *fuzzy*, o mas generalmente con metodologías propias del ámbito de la Soft Computing. Sin embargo esa forma de modelización

de la subjetividad, tan desarrollada en otros ámbitos, prácticamente no ha sido aplicada al caso del diseño de algoritmos heurísticos, a pesar de que todo apunta que este puede ser un enfoque muy prometedor porque, aparte de proporcionar soluciones tan cercanas al óptimo como las otras heurísticas convencionales ya conocidas,

a) encuentran la solución del problema en cuestión con un menor costo que los demás métodos,
b) como es habitual en el marco de las metodologías *fuzzy*, generalizan las heurísticas ya conocidas (que deben ser casos particulares de las aquí presentadas, en los valores 0 o 1 del grado de cumplimiento que se considere), y
c) la hibridación en el contexto de la Soft Computing favorece y enriquece la aparición de procedimientos originales que pueden ayudar a la resolución de nuevos problemas.

Pero, si bien la traza histórica de los conjuntos y los sistemas *fuzzy* es bien conocida, no pasa lo mismo con la de la Soft Computing, tan bien acogida ahora en muchos ambientes en los que hace solo unos cuantos años estaba proscrita. Por ello, y porque algo parecido le pasa al área de las heurísticas, también denostada durante mucho tiempo casi en los mismos círculos que lo era "lo *fuzzy*", en lo que sigue describiremos sucintamente que es la Soft Computing y que se entiende por heurística, y a partir de ambos conceptos, trataremos de encontrar una zona común en la que se pueda compaginar lo mejor de ambos mundos. El resultado será doble. En primer lugar surgirán los procedimientos metaheurísticos basados en Soft Computing, que se perfilan como una de las herramientas con un futuro mas prometedor de cara a la solución efectiva de problemas no resolubles hasta ahora, así como para encontrar soluciones a la medida de quien las busca. En segundo lugar, y como consecuencia de lo anterior, surgirá una nueva descripción de las componentes que definen la Soft Computing, que ampliará aún mas su ámbito de aplicaciones.

## 2. Soft Computing

Hasta que en 1994 L. A. Zadeh [2] dio la primera definición de "Soft Computing", la referencia a los conceptos que actualmente esta maneja solía hacerse de forma atómica, es decir, se hablaba de manera aislada de cada uno de ellos con indicación del empleo de metodologías *fuzzy*.

Aunque la idea de establecer el área de Soft Computing se remonta a 1990 [3], como se ha dicho fue en [2] donde L.A. Zadeh propuso la definición de Soft Computing, estableciéndola en los siguientes términos:

> "Básicamente, Soft Computing no es un cuerpo homogéneo de conceptos y técnicas. Mas bien es una mezcla de distintos métodos que de una forma u otra cooperan desde sus fundamentos. En este sentido, el principal objetivo de la Soft Computing es aprovechar la tolerancia que conllevan la imprecisión y la incertidumbre, para conseguir manejabilidad, robustez y soluciones de bajo costo. Los principales ingredientes de la Soft Computing son la Lógica *Fuzzy*, la Neuro-computación y el Razonamiento Probabilístico, incluyendo este último a los Algoritmos Genéticos, las Redes de Creencia, los Sistemas Caóticos y algunas partes de la Teoría de Aprendizaje. En esa asociación de Lógica *Fuzzy*, Neurocomputación y Razonamiento Probabilístico, la Lógica *Fuzzy* se ocupa principalmente de la imprecisión y el Razonamiento Aproximado; la Neurocumputación del aprendizaje, y el Razonamiento Probabilístico de la incertidumbre y la propagación de las creencias".

Queda claro así como la Soft Computing no esta definida precisamente, sino que en una primera aproximación se define por extensión, por medio de distintos conceptos y técnicas que intentan superar las dificultades que surgen en los problemas reales que se dan en un mundo que es impreciso, incierto y difícil de categorizar.

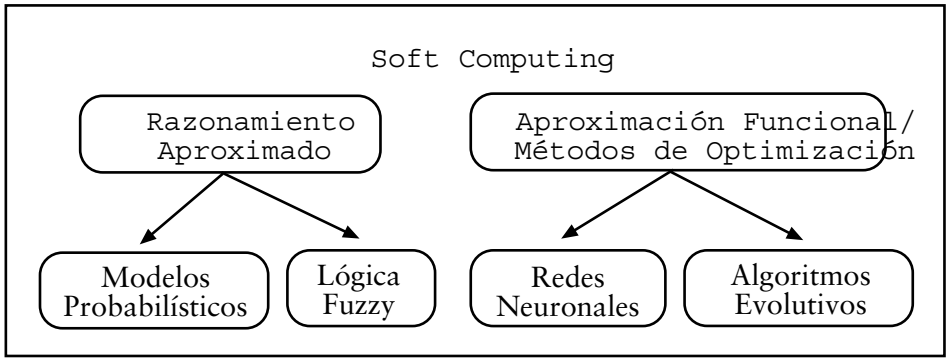
Aunque ha habido varios intentos de ajustar mas esta definición, no han sido muy fructíferos. Así, por ejemplo en [4], a la vista de la dificultad de dar una nueva definición del campo de una manera exacta y consensuada, y de la mayor sencillez de hacerlo por medio de sus características, los autores proponen la siguiente definición de trabajo, que vuelve a ser de tipo descriptivo: "Cualquier proceso de computación que expresamente incluya imprecisión en los cálculos en uno o mas niveles, y que permita cambiar (disminuir) la granularidad del problema o suavizar los objetivos de optimización en cualquier etapa, se define como perteneciente al campo de la Soft Computing".

El punto de vista que aquí consideramos, y que adoptamos en lo que sigue, implica otra forma de definir Soft Computing. Se trata de considerarla como antitesis de lo que podríamos denominar "Hard Computing", de manera que podría verse la Soft Computing como un conjunto de técnicas y métodos que permitan tratar las situaciones practicas reales de la misma forma que suelen hacerlo los seres humanos, es decir, en base a inteligencia, sentido común, consideración de analogías, aproximaciones, etc. En este sentido Soft Computing es una familia de métodos de resolución de problemas cuyos primeros miembros serían el Razonamiento Aproximado y los Métodos de Aproximación Funcional y de Optimización, incluyendo los de búsqueda. En este sentido, la Soft Computing queda situada como la base teórica del área de los Sistemas Inteligentes, y se hace patente que la diferencia entre el área de la Inteligencia Artificial clásica, y la de los Sistemas Inteligentes, es que la primera se apoya en

la denominada Hard Computing, mientras que la segunda lo hace en la Soft Computing"

Desde este otro punto de vista, en un segundo nivel, la Soft Computing se puede desgranar entonces en otros componentes que ayudan a una definición por extensión, como la que se dio en primer lugar. Desde el principio se han considerado que en ese segundo nivel los componentes mas importantes son el Razonamiento Probabilístico, la Lógica y los Conjuntos *Fuzzy*, las Redes Neuronales y los Algoritmos Genéticos [5] , que debido a su alta interdisciplinariedad, la importancia y relevancia de sus aplicaciones, y el volumen de resultados logrados, inmediatamente se destacaron de otras metodologías como, las ya citadas Teoría del Caos, Teoría de la Evidencia, etc.

Pero la mencionada popularidad de los Algoritmos Genéticos, junto con la eficacia demostrada en una gran variedad de áreas y aplicaciones, el intento de conseguir imitar a las criaturas naturales: plantas, animales, seres humanos, que evidentemente tienen una naturaleza "soft" (flexible, adaptable, creativa, inteligente, ...), y sobre todo sus extensiones y diferentes versiones, hicieron que ese cuarto ingrediente del segundo nivel pasara a ser el de los bien conocidos Algoritmos Evolutivos, que de esta forma se constituyeron en el cuarto integrante fundamental de la Soft Computing, como se refleja en el siguiente esquema,




Así, desde esta última concepción de Soft Computing, podemos hacer una descripción de otras áreas que surgen en torno a ella, sin mas que tener en cuenta algunas de las posibles combinaciones que pueden darse:

1. A partir del primer nivel, comenzando por los métodos de Razonamiento Aproximado, cuando nos concentramos solo en los Modelos Probabilísticos, nos encontramos con la Teoría de Dempster-Shafer

y las Redes Bayesianas. Pero cuando los Modelos Probabilísticos los consideramos combinados con la Lógica *Fuzzy*, e incluso con algunas otras lógicas multivaluadas, nos surgen los que podríamos denominar Modelos Probabilísticos Híbridos, fundamentalmente los de la Teoría de Probabilidad para Eventos *Fuzzy*, los de Creencias de Eventos *Fuzzy* y los Diagramas *Fuzzy* de Influencia.

2. Cuando nos concentramos en los mas importantes desarrollos directamente asociados a la Lógica *Fuzzy*, aparecen destacados los Sistemas *Fuzzy*, y en particular los Controladores *Fuzzy*. Entonces de la incidencia combinada de Lógica *Fuzzy* con las Redes Neuronales y Algoritmos Evolutivos, surgen los llamados Sistemas Híbridos basados en Lógica *Fuzzy*, cuyos mas destacados exponentes son los Sistemas Neuronales *Fuzzy*, los Controladores Ajustados mediante Redes Neuronales (Sistemas *Fuzzy* Neuronales, distintos de los mencionados Sistemas Neuronales *Fuzzy*) y los Controladores basados en Lógica *Fuzzy* generados y ajustados con Algoritmos Evolutivos.
3. Moviéndonos en el primer nivel a la otra gran área que abarca la Soft Computing, Aproximación Funcional/métodos de Optimización, el primer componente que aparece es el de las Redes Neuronales, y por tanto los diversos modelos de las mismas (Recurrentes, "Feedforward", Perceptrón, Hopfield, ...). En este contexto, la interacción con las metodologías de la Lógica *Fuzzy* y los Algoritmos Evolutivos llevan a lo que suelen llamarse Sistemas Neuronales Híbridos, particularmente el control *fuzzy* de parámetros de Redes, y la generación formal y la generación de pesos en Redes Neuronales
4. La cuarta componente típica de la Soft Computing, por otro lado la que quizás tenga una historia mas corta, pero posiblemente la de mayor actualidad, es la constituida por los Algoritmos Evolutivos. Asociados a los mismos aparecen cuatro grandes áreas mas importantes: Las Estrategias Evolutivas y la Programación Evolutiva, y los Algoritmos Genéticos y la Programación Genética. Si no atendiéramos mas que a estas últimas áreas, podríamos considerar que en este caso la amalgama de metodologías y técnicas asociadas a la Soft Computing desembocan en tres importantes líneas: los Sistemas Genéticos *Fuzzy* , los Sistemas Bioinspirados y las Aplicaciones del Control *Fuzzy* sobre parámetros evolutivos.

Concentrándonos un poco mas en esta última componente, hemos de hacer algunas consideraciones adicionales. En primer lugar que, independientemente de la amplitud de miras con que se contemple lo que pueden abarcar los Sistemas Genéticos *Fuzzy*, los Sistemas Bioinspirados

y las Aplicaciones del Control *Fuzzy* sobre parámetros evolutivos, en esa descripción se echan en falta importantes tópicos de investigación y desarrollo. En segundo lugar, que si nos referimos mas particularmente a los Sistemas Bio-Inspirados, resulta patente que no solo son fruto de la Lógica *Fuzzy*, las Redes Neuronales o los Algoritmos Evolutivos, con todas las variantes que para estas tres componentes podamos considerar, sino que en sus definiciones, implementaciones y desarrollos también intervienen otras metodologías de una gran importancia.

Por eso, en lo que sigue vamos a justificar una nueva definición de las componentes de la Soft Computing, preliminarmente comentada en [6], de modo que sin perder esta su esencia, podamos tener una perspectiva mas clara de las diferentes áreas que abarca.

## 3. Heurísticas y Metaheurísticas

Como se puso de manifiesto en [7], desde el *Fuzzy Boom* de los 90, las metodologías basadas en conjuntos *fuzzy*, la Soft Computing, se han instalado de manera permanente en todas las áreas de la investigación, el desarrollo y la innovación. Sus aplicaciones se multiplican en todos los ámbitos de nuestra vida cotidiana: salud, banca, hogar, ... y son objeto de estudio en los diferentes niveles educativos. Por otro lado, no cabe duda de que gracias al potencial tecnológico del que disponemos hoy día, los computadores pueden abordar problemas de formidable complejidad, tanto en comprensión como en dimensión, en una gran variedad de campos nuevos (Bioinformática, Minería de Datos, Ingeniería del Conocimiento,...).

Como anteriormente se comentó, desde mediados de los 90, los Algoritmos Genéticos, o desde un punto de vista general los Algoritmos Evolutivos, se están mostrando como métodos muy valiosos para encontrar buenas soluciones a problemas concretos en estos campos. Como fruto de su atractivo científico, de la diversidad de sus aplicaciones y de la notable eficacia de sus soluciones en el contexto de los Sistemas Inteligentes, se han incorporado al segundo nivel de las componentes de la Soft Computing.

Los Algoritmos Evolutivos sin embargo no son mas que una clase mas de Heurísticas, o de Metareurísticas, como también lo son la Búsqueda Tabú, el Enfriamiento (Recocido) Simulado, los métodos de Escalada, la Búsqueda por Entornos Variables, los Algoritmos de Estimación de Distribuciones (EDA), la Búsqueda Dispersa, los GRASP, la Búsqueda Reactiva, y muchos mas. Generalmente, todos estos algoritmos heurísticos

(metaheurísticas) suelen proporcionar soluciones que no son las optimas, pero que satisfacen en buena medida al decisor o usuario. Cuando estos actúan desde el principio de que es mejor satisfacer que optimizar, le dan perfecto sentido en este contexto a la famosa frase de Zadeh: "...en contraste con la computación tradicional (hard), la Soft Computing se beneficia de la tolerancia asociada a la imprecisión, la incertidumbre, y las verdades parciales para conseguir tratabilidad, robustez, soluciones de bajo costo y mejores representaciones de la realidad". Consiguientemente, entre las componentes de la Soft Computing, en lugar de los Algoritmos Evolutivos, que pueden representar solo una parte de los métodos de búsqueda y optimización que se emplean, deberían considerarse los Algoritmos Heurísticos o aún mejor las Metaheurísticas.

Suele haber mucha controversia y discusión acerca de la diferencia entre Metaheurística y Heurística. No es nuestra intención aquí entrar en ese debate, pero si nos interesa hacer una mínima reflexión acerca de ambos conceptos. Como es de sobra conocido, el término heurística proviene de la palabra griega "heuriskein", cuyo significado está relacionado con el concepto de encontrar y se vincula a la famosa y supuesta exclamación ¡eureka! de Arquímedes.

Con ese origen se han desarrollado un gran número de procedimientos heurísticos para la resolución de problemas de optimización específicos con mucho éxito, de los que se intenta extraer lo mejor de ellos y emplearlo en otros problemas o en contextos más extensos. Esto ha contribuido al desarrollo científico de este campo de investigación y a extender la aplicación de sus resultados. Surgen así las denominadas metaheurísticas, término que apareció por primera vez en un artículo de Fred Glover en 1986.

El término metaheurística deriva de la composición de la palabra heurística, ya comentada, con la del sufijo meta (más allá o de nivel superior). Aunque no existe una definición formal de qué es una metaheurística, las dos siguientes propuestas dan una representación clara de la noción general del término.

a) Osman y Laporte la definen así: una metaheurística se define formalmente como un proceso iterativo que guía una heurística subordinada, combinando de forma inteligente diferentes conceptos para explorar y explotar el espacio de búsqueda.
b) según Voss et al.: una metaheurística es un proceso maestro iterativo que guía y modifica las operaciones de heurísticas subordinadas para producir, de forma eficiente, soluciones de alta calidad. En cada iteración, puede manipular una solución (completa o incompleta) o

un conjunto de soluciones. Las heurísticas subordinadas pueden ser procedimientos de alto o bajo nivel, o simplemente una búsqueda local o método constructivo.

Parece por tanto claro que el concepto de metaheurística tiene un carácter mas generalista que el de heurística. Por eso en lo que sigue nos concentramos en las primeras, comenzando por puntualizar que, en los términos que las hemos definido, una metaheurística será mejor que otra simplemente en función del rendimiento que proporcionen a la hora de resolver problemas.

Para conseguir el mejor rendimiento de las metaheurísticas, es deseable que tengan una serie de "buenas propiedades", entre las que se encuentran las siguientes [8]:

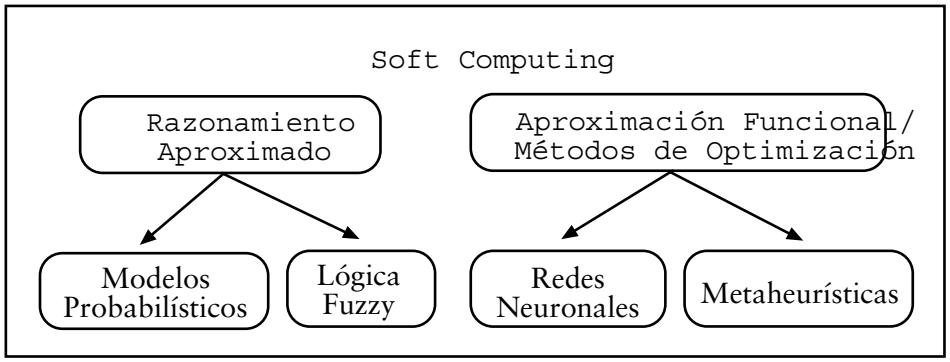
1) Simplicidad, ya que una metaheurística debe estar basada en un principio sencillo y claro que la haga fácil de comprender,
2) Independencia, puesto que no puede depender del marco o del agente tecnológico en el que se vaya a desarrollar.
3) Coherencia, porque los elementos que caractericen la metaheurística deben derivarse de forma natural de los principios que la inspiran.
4) Efectividad, ya que los algoritmos particulares que se produzcan a partir de las metaheurísticas deben proporcionar soluciones óptimas o muy cercanas a las óptimas, es decir, de alta calidad en algún sentido propio que cada usuario deberá especificar, y que podrá depender de cada problema concreto.
5) Eficacia, en el sentido de fallar solo en ocasiones muy raras ante casos prácticos del mundo real.
6) Eficiencia, como una característica, mas que deseable, exigible en términos de recursos, es decir, de tiempo de ejecución, de espacio de memoria y, en definitiva, de costos de desarrollo.
7) Generalidad, de forma que se pueda utilizar provechosamente en una gama de situaciones y problemas tan amplia como sea posible.
8) Adaptabilidad, para que pueda adecuarse a los diferentes contextos de aplicación y a los distintos casos que se consideran.
9) Robustez, ya que no puede ser muy sensible a pequeñas alteraciones del modelo o contexto de aplicación.
10) Interactividad, para que el decisor pueda mejorarla a partir de su experiencia y conocimientos.
11) Diversidad, para que se permita al usuario elegir entre las distintas soluciones alternativas que proporcione la metaheurística, y

12) Autonomía, para facilitar su funcionamiento automático global, o al menos en alguna de las facetas que la caractericen.

A la vista de su definición y de ésta serie de características deseables, es obvio que entre las metaheurísticas, como no podía ser de otra forma, se encuentran los Algoritmos Evolutivos, y que por tanto tienen un fácil acomodo con las demás componentes del segundo nivel de la Soft Computing, con las que sus sinergias deben facilitar la aparición de nuevas metodologías, esquemas y marcos teóricos y prácticos que, como se explica en [3], ayuden a entender y tratar mejor la generalizada imprecisión del mundo real.

## 4. Una revisión de las componentes de la Soft Computing

Retomando en este punto la anterior descripción y análisis de las componentes que describen la Soft Computing en los distintos niveles, podríamos concretar que, en el segundo, sus componentes mas importantes son el Razonamiento Probabilístico, la Lógica y los Conjuntos *Fuzzy*, las Redes Neuronales y, a la vista de lo explicado, las Metaheurísticas, que típicamente englobarían los Algoritmos Evolutivos, pero no quedarían circunscritas a estos exclusivamente. Con esto, el nuevo marco definitorio de las principales metodologías que integran la Soft Computing quedaría como se describe en el siguiente esquema




Como antes hemos tenido ocasión de explicar, las metodologías que integran la Soft Computing, mas que poder ser entendidas de forma aislada, hay que comprenderlas como el resultado de la cooperación, la asociación, la complementariedad o la hibridación de sus componentes de segundo nivel. En esta línea, tiene perfecto sentido que intentemos ahora

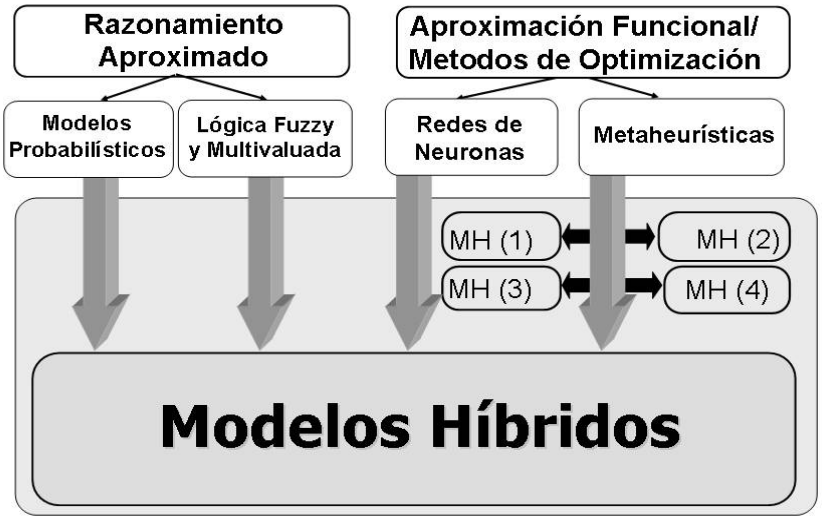
ver cuales son las nuevas facetas teórico-prácticas que se derivan de la aparición entre esas componentes de las metaheurísticas.

Las metaheurísticas disponibles son tantas y tan variadas, que es prácticamente imposible ponerse de acuerdo en una forma de clasificarlas que sea universalmente aceptada. No obstante, la jerarquía sobre la que hay mas consenso considera tres (cuatro) grupos mas destacados:

1) las metaheurísticas para los procedimientos evolutivos, basadas en conjuntos de soluciones que evolucionan según principios de la evolución natural,
2) las metaheurísticas para los métodos de relajación, que son métodos de solución de problemas que utilizan adaptaciones del modelo original que son más sencillas de resolver,
3) las metaheurísticas para las búsquedas por entornos, que recorren el espacio de soluciones explotando estructuras de entornos asociadas a esas soluciones, y
4) otros tipos de metaheurísticas que se corresponden con tipos intermedios entre los anteriores, o derivados en algún sentido de estos, y que por su gran variabilidad, para evitar la dispersión, no consideraremos en lo que sigue.

Bien, haciendo nuestra esta forma de clasificar las metaheurísticas, lo primero que podemos observar es que la anterior definición de Soft Computing que hemos hecho "por extensión" en función de sus componentes, no solo mantiene la esencia de la definición original de L.A. Zadeh, sino que la generaliza y amplia para contemplar nuevas posibilidades. En efecto, si esos cuatro grupos de metaheurísticas los nombramos MH(1), ... MH(4), respectivamente, el anterior diagrama descriptivo de las principales metodologías que integran la Soft Computing, podría representarse ahora de forma mas explicita como se muestra a continuación, donde, debido a que las componentes clásicas de la Soft Computing (Modelos Probabilísticos, Logica *Fuzzy*, Redes de Neuronas y Algoritmos Evolutivos) siguen estando ahí, las diferentes áreas conocidas y estudiadas, se mantienen como hasta ahora, surgiendo como siempre cuando dos o mas de esas componentes se interrelacionan entre si. Sin embargo, como consecuencia de haber incorporado nuevas posibilidades en la cuarta componente (metaheurísticas) ahora tiene perfecto sentido esperar que aparezcan nuevos Modelos Híbridos que desarrollar.

Para mostrar el abanico de áreas de estudio con el que podemos contar, cuando la componente base que se toma es la de las Metaheurísticas, en lo que sigue nos concentraremos en describir las hibridaciones que surgen empleando para ello la anterior categorización.

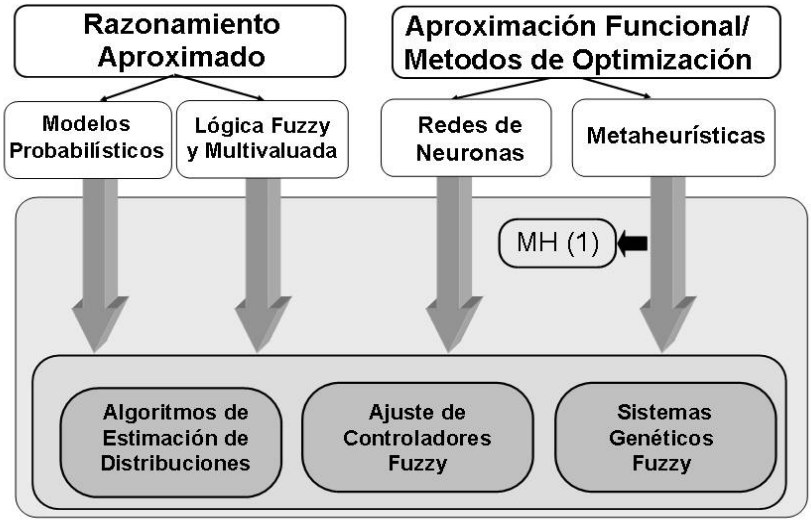



**5. Metaheurísticas Híbridas en Soft Computing**

Consideremos los cuatro grandes grupos de metaheurísticas que mas arriba hemos referido. A partir de ellos, en lo que sigue describiremos las nuevas metaheuristicas que surgen, deteniéndonos brevemente solo en aquellas que por su novedad tengan un nivel de desarrollo o popularidad menor.

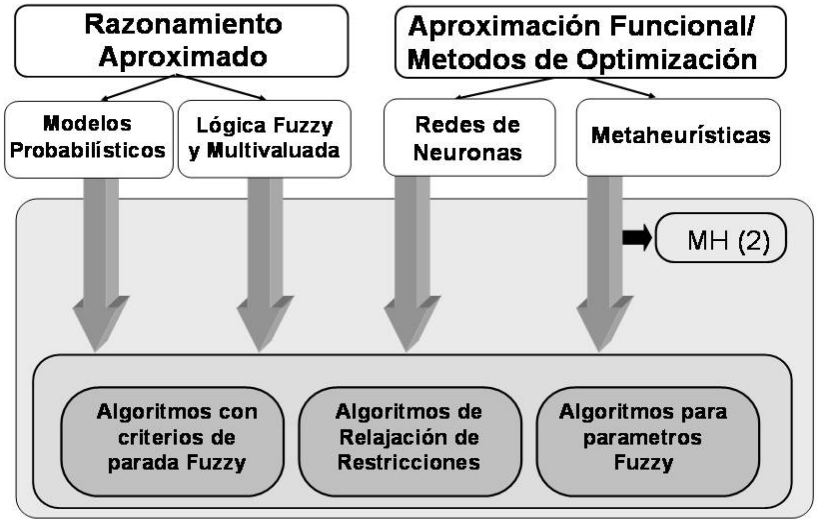
5.1. Supongamos las metaheurísticas evolutivas. Estas metaheurísticas, las mas populares con mucha diferencia sobre todas las demás, definen mecanismos para que se desarrolle una evolución en el espacio de búsqueda de los conjuntos de soluciones, con la finalidad de aproximarse a la solución óptima con los elementos que van sobreviviendo en las sucesivas generaciones de poblaciones. En el contexto de la Soft Computing las hibridaciones que toman como referencia estas metaheurísticas son fundamentales.

Sin embargo, una mínima descripción de esta importantísima y amplia área, en la que se incardinan desde los Sistemas Genéticos *Fuzzy*, hasta el ajuste de controladores *fuzzy* con algoritmos evolutivos, pasando por los Algoritmos de Estimación de Distribuciones, los Sistemas Bioinspirados, etc., escapa por completo del ámbito de este articulo y se aleja de su finalidad descriptiva y discursiva, por lo que se remite al lector interesado a fuentes bibliográficas recientes bien contrastadas ([9,10,11]).




5.2. Metaheurísticas de relajación. Se obtiene una relajación de un problema real cuando este se simplifica eliminando, debilitando o modificando algún elemento característico del mismo. Las metaheurísticas de relajación son estrategias para el empleo de relajaciones del problema en el diseño de heurísticas, que persiguen encontrar una solución para un problema que, de no emplear esta metodología, sería muy difícil poderlo resolver. Ejemplos triviales de las mismas son los redondeos o los ajustes de naturaleza, como ocurre cuando a una cantidad expresada imprecisa y lingüísticamente, se le asocia un valor numérico exacto. Desde este punto de vista, una alternativa real es la de flexibilizar los algoritmos exactos, introduciendo criterios de parada *fuzzy*, lo que finalmente conduce a metaheurísticas de relajación basadas en reglas; admitiendo la vaguedad de los coeficientes, justificando los algoritmos para resolver problemas con parámetros *fuzzy*; y relajando la verificación de las restricciones, permitiendo ciertas violaciones en su cumplimiento.

Por ilustrar un poco mas concretamente alguna de estas metaheurísticas, consideramos los algoritmos con criterios de parada *fuzzy* [12,13]. Como se sabe, los criterios de parada fijan las condiciones de finalización del procedimiento iterativo de un algoritmo, estableciéndose dichos criterios a partir de las características teóricas del problema, del tipo de solución que se busca y del tipo del algoritmo que utilice. Si un algoritmo dado proporciona la sucesión $(x_n)$ de soluciones factibles, algunos de los criterios de parada más frecuentes son:




a) parar el proceso después de N iteraciones,
b) parar el proceso cuando la distancia relativa o absoluta entre dos elementos de la sucesión a partir de una determinada iteración sea menor o igual que un valor prefijado,
c) parar el proceso cuando una medida prefijada $g(x_n)$ satisfaga una determinada condición como por ejemplo el de ser menor o igual que una constante.

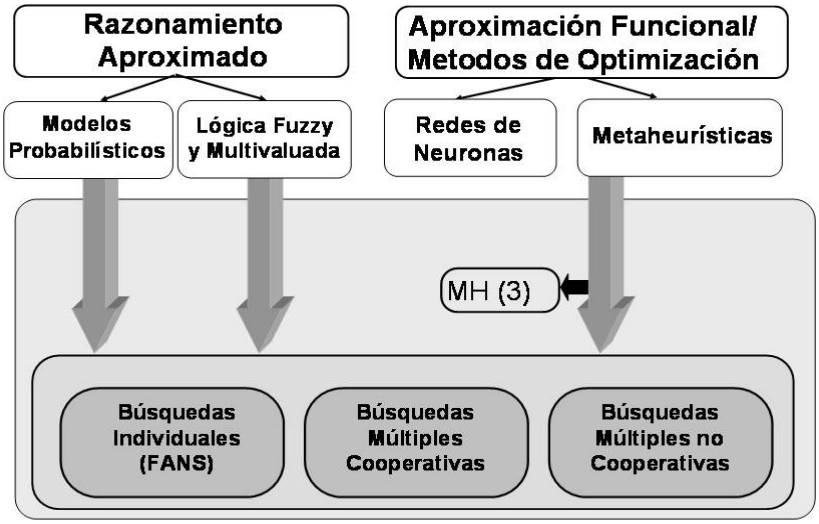
En resumen, se puede decir que un algoritmo determina un conjunto de referencia y se detiene cuando se ha obtenido el conjunto especificado en los criterios de parada.

Por tanto, la flexibilización de los algoritmos exactos con la introducción de criterios de parada *fuzzy* supone considerar que el conjunto de referencia es un conjunto *fuzzy*. Y los criterios de parada *fuzzy* se fijan en función del grado de pertenencia de los elementos.

5.3. Metaheurísticas de búsqueda. Probablemente estas sean las metaheurísticas mas importantes, por generales. Su funcionamiento básico consiste en establecer estrategias para recorrer el espacio de soluciones del problema transformando las soluciones de partida de forma iterativa. Aunque de entrada pudieran parecer similares a las búsquedas evolutivas, no lo son debido a que aquellas basan su funcionamiento en la evolución de una población de individuos sobre el espacio de búsqueda.

La descripción de estas metaheurísticas suele hacerse a partir de una diversidad de metáforas, que pueden servir para justificarlas como bio-inspiradas, de carácter sociológico, basadas en la Naturaleza, ... que las confieren un alto grado de popularidad.

Ahora bien, fuera de ese marco descriptivo, dado que una búsqueda puede realizarse mediante un único procedimiento buscador, o por mas de uno, y que en este caso, los métodos de búsqueda podrían cooperar entre si o no, las metaheurísticas de búsqueda (sin que esta clasificación sea exclusiva de este apartado) pueden considerarse como individuales o múltiples, permitiéndose en este último caso la posibilidad de que los diferentes agentes cooperen entre si o no. Las distintas opciones que pueden surgir en el contexto de la Soft Computing se recogen en el siguiente esquema.




Entre las metaheurísticas individuales mas conocidas se encuentran el "Hill Climbing", las de tipo "Greedy", las de Arranque Múltiple, las de Entorno Variable, las de Recocido Simulado o las Búsquedas Tabú, que pueden tener sus propias extensiones *fuzzy*.

Independientemente de su forma concreta de actuación, todas estas metaheurísticas progresan por el espacio de búsqueda en base a evaluaciones de la función objetivo del problema concreto que se esté tratando de resolver, lo que explícitamente supone realizar valoraciones numéricas, con ayuda de una función objetivo, en un espacio exactamente determi-

nado. Pero con mucha frecuencia, la función objetivo representa alguna propiedad vagamente establecida, y el espacio de búsqueda, o los entornos en los que ésta se va realizando, no tiene unas fronteras nítidamente definidas, lo que da perfecto sentido a enfocar la aplicación de estas metaheurísticas con elementos teóricos del campo de los conjuntos y la lógica *fuzzy*. Precisamente en ese contexto surgen los algoritmos de tipo FANS (*Fuzzy Adapative Neighborhood Search*) [14,15].

FANS es un método de búsqueda por entornos donde las soluciones se evalúan no solo en términos de la función objetivo, sino también mediante el empleo de propiedades y conceptos *fuzzy* que permiten valoraciones cualitativas sobre las soluciones. Además, es un método adaptable al contexto, ya que su comportamiento varia en función del estado de la búsqueda a través del uso de varios administradores o "schedulers".

FANS se base en cuatro componentes principales: un operador (OP), para construir soluciones; una propiedad *fuzzy* (FC) para cualificarlas y evaluarlas; un administrador de operación (OS), para adaptar el comportamiento o características del operador; y un administrador de vecindario (NS), para generar y seleccionar una nueva solución.

A continuación se muestra un esquema del algoritmo, en el que se puede apreciar la iteración entre las cuatro componentes antes descritas

```
SolAct: Solución actual
OS: Administrador de Operador
NS: Administrador de Vecindario
OP: Operador de modificación
FC: Concepto Difuso
MejorSol: Mejor Solución encontrada
WhileDo(!condición de finalización)
   vecino = NS->Ejecutar(SolAct, FC, OP, excepción);
   If(hay condición de excepción)
        Then OP = OS->Ejecutar(Estadísticas);
        Else SolAct = vecino;
             If(f(SolAct) > f(MejorSol))
                  Then MejorSol = SolAct;
             Fi
   Fi
   If(condiciones de adaptación)
     FC->AdaptarParametros() ;
   Fi
   iter++;
Od
Reportar MejorSol ;
```

Como puede comprobarse, después de una etapa de inicialización el algoritmo itera hasta que se verifica una cierta condición de finalización. En cada iteración se ejecuta el administrador de vecindario NS, el cual utilizando la solución actual SolAct, el concepto *fuzzy* FC y el operador OP, generará cierto número de soluciones vecinas, las cualificara mediante FC y elegirá alguna en función de algún criterio. Si es posible, el administrador devuelve una solución vecina, que pasa a ser la solución actual. Además se realiza la comparación correspondiente para verificar si es la mejor de todas las visitadas. Si, por ejemplo, no existieran soluciones vecinas satisfaciendo el criterio utilizado, se produciría una condición de excepción. Como respuesta, se ejecutaría el administrador de operación OS, que modificaría en algún sentido el operador utilizado. En la próxima iteración, el administrador de vecindario dispondría de un operador diferente para buscar soluciones. Finalmente, si se verifican ciertas condiciones, la propiedad difusa que se está empleando se adapta, lo que ocurre por ejemplo cada vez que la solución actual cambia, o después de un número fijo de iteraciones.

Por otro lado, si el procedimiento de búsqueda se realiza a partir de varias metaheurísticas, como se ha comentado anteriormente, tenemos la posibilidad de que estas cooperen entre si o no [16], y por tanto la generalización de todo lo descrito hasta ahora al contexto del paralelismo, algo que obviamente escapa del ámbito de este trabajo, pero sobre el que es interesante hacer una mínima reflexión, ya que con la proliferación de la computación paralela, las cada vez mas potentes estaciones de trabajo, y la velocidad de las redes de comunicación, las implementaciones paralelas de las metaheurísticas han surgido como algo natural, proporcionando una interesante alternativa para aumentar la velocidad de la búsqueda de las soluciones. En ese sentido se han propuesto y aplicado diversas estrategias que se han demostrado muy eficientes para resolver problemas de gran tamaño y para encontrar soluciones mejores que las de sus contrapartidas secuenciales, debido a la división del espacio de búsqueda, o porque han mejorado la intensificación y la diversificación de la búsqueda. Por ello, el paralelismo, y por tanto las metaheurísticas múltiples, no sólo constituyen una vía para reducir los tiempos de ejecución de las metaheurísticas individuales, sino también para mejorar su efectividad y robustez.

En el ámbito de la Soft Computing, la idea básica que hasta ahora se ha desarrollado ha consistido en suponer que se dispone de un conjunto de agentes resolvedores [17], cuya función básica es ser algoritmos de solución de problemas de optimización combinatoria, y ejecutarlos de forma cooperativa a través de un agente coordinador para resolver el

problema en cuestión, teniendo como premisa fundamental la generalidad basada en un conocimiento mínimo del problema. Cada agente resolvedor actúa de forma autónoma y se comunica solamente con un agente coordinador para enviarle las soluciones que va encontrando y para recibir de este las directivas que le indiquen cómo seguir actuando. El agente coordinador recibe las soluciones encontradas por cada agente resolvedor para el problema, y siguiendo una base de reglas *fuzzy* que modelan su comportamiento, crea las directivas que envía a éstos, llevando de esta manera todo el control de la estrategia.

## Conclusión

El concepto de conjunto *fuzzy* ha sido y es un paradigma en el mundo científico-tecnológico, de relevantes repercusiones en todos los sectores sociales a causa de la diversidad de sus aplicaciones, de la facilidad de su transferencia tecnológica, y del ahorro económico que su uso supone. Aunque cuando se publicó el primer articulo sobre el tema, hace aproximadamente cuarenta años, hubo sectores académicos muy reacios al mismo, el tiempo ha demostrado que los conjuntos fuzzy constituyen el núcleo de un cuerpo doctrinal de indudable solidez, dinamismo y reconocimiento internacional que se denomina Soft Computing.

Es precisamente ese dinamismo el que nos ha llevado en este articulo a reflexionar sobre cuales son los limites definitorios de la Soft Computing, intentando ampliar el abanico de sus componentes básicas al incorporar a las mismas las Metaheurísticas. Esta perspectiva mas amplia y general de la Soft Computing permite la posibilidad de incorporar nuevos esquemas de búsqueda/ optimización aún no desarrollados, sin que ninguno de los ya explorados sea protagonista, evitando de esta forma, como L.A. Zadeh indicó en [3], la tendencia a proclamar que aquella metodología que a nosotros nos interesa es la mejor de todas (una versión mas, como el mismo Zadeh señaló también, del famoso Principio del Martillo: "Cuando la única herramienta que tienes es un martillo, todo lo que te encuentras son clavos")

## Referencias

[1] Zadeh, L.A. (1965): Fuzzy Sets. *Information and Control,* 338-353.
[2] Zadeh, L.A. (1994). Soft Computing and Fuzzy Logic. *IEEE Softwar*e 11, 6, 48-56.
[3] Zadeh, L.A. (2001): Applied Soft Computing. *Applied Soft Computing* 1, 1–2

[4] Li, X., Ruan, D. and van der Wal, A.J. (1998): Discussion on soft computing at FLINS'96. *International Journal of Intelligent Systems*, 13, 2-3, 287- 300.
[5] Bonissone, P (2002): Hybrid Soft Computing for Classification and Prediction Applications. Conferencia Invitada. *1st. International Conference on Computing in an Imperfect World* (Soft-Ware 2002), Belfast
[6] Verdegay, J.L. (2005): Una revisión de las metodologías que integran la "Soft Computing". *Actas del Simposio sobre Lógica Fuzzy y Soft Computing* (LFSC2005). Granada, 151-156
[7] Verdegay, J.L., Ed. (2003): *Fuzzy Sets-based Heuristics for Optimization*. Studies in Fuzziness. Springer Verlag
[8] Melián, B., Moreno Pérez, J.A., Moreno Vega, J.M. (2003): Metaheurísticas: Una visión global. *Revista Iberoamericana de Inteligencia Artificial* 19, 2, 7-28
[9] Cordón, O., F. Gomide, F. Herrera, F. Hoffmann, L. Magdalena (2004): Ten Years of Genetic Fuzzy Systems: Current Framework and New Trends. *Fuzzy Sets and Systems* 141:1, 5-31.
[10] Larrañaga, P., J.A. Lozano, H. Mühlenbein (2003): Algoritmos de estimación de distribuciones en problemas de optimización combinatoria. *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial*, 19(2), 149-168.
[11] Arenas, M.G., F. Herrera, M. Lozano, J.J. Merelo, G. Romero, A.M. Sánchez (Eds) (2005): *Actas del IV Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB'05) I y II.*
[12] Vergara-Moreno, E (1999): *Nuevos Criterios de Parada en Algoritmos de Optimizacion*. Tesis Doctoral. Universidad de Granada.
[13] Verdegay, J.L. y E. Vergara-Moreno (2000): Fuzzy Termination Criteria in Knapsack Problem Algorithms. *Mathware and Soft Computing* VII, 2-3, 89-97.
[14] Pelta, D.A. (2002): *Algoritmos Heurísticos en Bioinformática* (2002). Tesis Doctoral. Universidad de Granada.
[15] Blanco, A., D. Pelta y J.L. Verdegay (2002): A Fuzzy Valuation-based Local Search Framework for Combinatorial Problems. *Fuzzy Optimization and Decision Making* 1, 177-193.
[16] Cruz Corona, C. (2005): *Estrategias cooperativas multiagentes basadas en Soft Computing para la solución de problemas de optimización*. Tesis Doctoral. Universidad de Granada.
[17] Pelta, D.A., A. Sancho-Royo, C. Cruz y J.L. Verdegay: Using memory and fuzzy rules in a co-operative multi-thread strategy for optimization. *Information Science* (en prensa)