Un prototipo del coordinador de un sistema metaheurístico cooperativo para el problema de la Mochila

Jose M. Cadenas•, M^a Carmen Garrido•, Vicente Liern°, Enrique Muñoz•, Emilio Serrano•

Resumen— **El problema algoritmo-instancia dice que conociendo un algoritmo y unos valores para sus parámetros que tengan muy buen comportamiento frente a una instancia del problema, es posible que ni este algoritmo ni estos parámetros funcionen bien para otra instancia del mismo problema. Esto conduce a la utilización de diferentes algoritmos metaheurísticos bajo un mismo esquema coordinado para resolver problemas de optimización combinatoria. La robustez se obtendrá por el uso de diferentes combinaciones de metaheurísticas que cooperan entre sí, alcanzando soluciones de muy alta calidad para diferentes clases de instancias.**

En este esquema, uno de los problemas que surge es el diseño del coordinador para realizar, de manera eficiente y/o efectiva, la cooperación entre las metaheurísticas. Se describirá el modelo de dicho coordinador y mediante un proceso de extracción de conocimiento se diseñará y definirá dicho coordinador.

El producto final obtenido es un prototipo para el problema de la mochila, utilizando un algoritmo genético, búsqueda tabú, temple simulado y colonia de hormigas, como conjunto de metaheurísticas a cooperar.

Palabras clave— **Sistemas Metaheurísticos Cooperativos, Análisis Inteligente de Datos, Extracción de Conocimiento, Sistema de reglas Fuzzy**

I. Introducción

En los problemas de optimización, generalmente, buscamos la mejor solución (o una suficientemente buena) a un problema entre muchas alternativas. Los problemas de optimización se suceden en nuestra vida diaria en cualquier lugar. Cada uno de nosotros constantemente resolvemos problemas de optimización tales como: encontrar el camino más corto desde nuestra casa al trabajo sujeto a las restricciones del tráfico u organizar nuestra agenda. Y los humanos encontramos soluciones a estos problemas diarios de una forma eficiente. ¿Por qué somos capaces de hacerlo? Porqué son problemas todavía tratables, tienen una dimensión lo suficientemente pequeña como para procesarlos. Sin embargo, estos problemas pueden crecer a grandes escales, tal como por ejemplo hacer más beneficioso el uso de una flota de aviones para reducir el costo de combustible. Entonces estos tipos de problemas son de tan alta dimensión y complejidad que se requieren de algoritmos computacionales para abordarlos.

En este tipo de problemas, en los cuales el dominio es típicamente finito, puede parecer trivial encontrar su solución ya que existe siempre un procedimiento elemental para determinar la solución óptima buscada, que es realizar una exploración exhaustiva del conjunto de soluciones. O sea, basta realizar los siguientes pasos: generar todas las soluciones factibles, calcular el costo asociado a cada una, y finalmente elegir la que haya obtenido el mejor valor. Sin embargo, aunque teóricamente este método siempre llega a la solución buscada, no es eficiente, pues el tiempo de cálculo necesario crece exponencialmente con el número de ítems del problema y por lo tanto procesarlas todas para encontrar la mejor entre ellas resulta prácticamente imposible aun para instancias de tamaño moderado.

Para mejorar la exploración exhaustiva aparecen las heurísticas. En Inteligencia Artificial se emplea el calificativo heurístico, en un sentido muy genérico, para aplicarlo a todos aquellos aspectos que tienen que ver con el empleo de conocimiento en la realización dinámica de tareas. Se habla de heurística para referirse a una técnica, método o procedimiento inteligente de realizar una tarea que no es producto de un riguroso análisis formal, sino de conocimiento experto sobre la tarea. En especial, se usa el término heurístico para referirse a un procedimiento que trata de aportar soluciones a un problema con un buen rendimiento, en lo referente a la calidad de las soluciones y a los recursos empleados.

Y para mejorar las heurísticas surgen las metaheurísticas que se pueden definir precisamente como, [6]:

Estrategias inteligentes para diseñar o mejorar procedimientos heurísticos muy generales con un alto rendimiento

Los algoritmos metaheurísticos brindan estrategias efectivas para encontrar soluciones aproximadas a problemas de optimización. Al trabajar con ellos, no es extraño encontrarnos con el problema algoritmo-instancia o problema de la selección del algoritmo, [7]. Este problema consiste en determinar qué algoritmo se debe seleccionar para resolver una instancia dada, de modo que se maximice una medida de rendimiento. Por lo general, el enfoque más común consiste en medir el rendimiento de un

• Departamento de Ingeniería de la Información y las Comunicaciones. Facultad de Informática. Universidad de Murcia. Spain. E-mail: jcadenas@um.es, mgarrido@dif.um.es, jayawata@hotmail.com, emilioserra@gmail.com

° Departamento de Matemáticas para la Economía y la Empresa. Universidad de Valencia. Spain. E-mail: Vicente.Liern@uv.es

conjunto de algoritmos sobre un conjunto de instancias, y utilizar aquel que se presente el mejor. Este esquema, denominado "winner-take-all", ha impulsado el desarrollo y refinamiento de algoritmos, pero ha impedido el desarrollo de otros, que si bien en promedio no son competitivos, pueden ofrecer un rendimiento excelente en instancias particulares del problema. En todo caso, como se indica en [4], el "problema de la selección automática del algoritmo es indecidible", lo que no niega la posibilidad de inducir un sistema que sugiera o aplique directamente un algoritmo, o conjunto de algoritmos, a partir de la evidencia teórico-experimental conseguida de la ejecución de diferentes algoritmos sobre conjuntos de instancias con diferentes características.

La cuestión ahora es: con qué medios vamos a intentar resolver la selección del mejor o mejores algoritmos. Pasemos a ver un posible caso en el uso de metaheurísticas.

En la figura 1 podemos apreciar dos metaheurísticas que se mueven por el espacio de búsqueda para encontrar la solución a una instancia dada. Si únicamente nos fijamos en el comportamiento que presentan las metaheurísticas desde su inicio hasta los puntos marcados con círculos concéntricos, ¿cuál de las metaheurísticas diríamos que se está comportando mejor?.

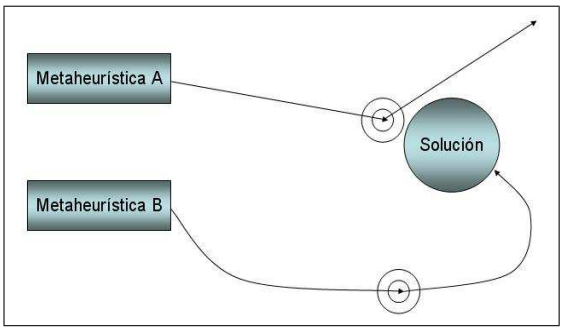

Fig. 1. Un posible caso de uso de metaheurísticas

A priori, de tener que elegir, nos quedaríamos con la *metaheurística A*. Debido a que es la que está más cercana a la solución, mientras que la B parece perdida.

Sin embargo, contemplando el resultado final resulta que, al contrario de lo que podía pensarse, es la *metaheurística A* la que no obtiene la solución, mientras que la B, tras un largo rodeo (indicando mayor tiempo de ejecución), alcanza la solución.

En [3] se plantea una esquema consistente en ejecutar las dos metaheurísticas paralelamente de manera que cada una modifique el comportamiento de la otra de forma eficiente. De esta manera podríamos obtener mejores soluciones en menos tiempo, es decir, mejorar su rendimiento.

De forma más general, podemos pensar en utilizar diferentes metaheurísticas bajo un mismo esquema coordinado para resolver problemas de optimización combinatoria. La robustez se obtendrá por el uso de diferentes combinaciones de metaheurísticas que cooperan entre sí, alcanzando soluciones de muy alta calidad para diferentes clases de instancias, [3].

Por tanto, en la siguiente sección vamos a presentar un esquema de metaheurísticas cooperativas. En este esquema, uno de los problemas que surge es el diseño de coordinador para realizar, de manera eficiente y/o efectiva, la cooperación entre las metaheurísticas. En la sección III se describirá el modelo de dicho coordinador, pasando a la sección IV la cual, mediante un proceso de extracción de conocimiento, diseñará y definirá dicho coordinador. En esta sección, se explicará todo el proceso realizado obteniendo un prototipo para el problema de la mochila, utilizando un algoritmo genético, búsqueda tabú, temple simulado y colonia de hormigas, como conjunto de metaheurísticas a cooperar. Para acabar presentaremos las conclusiones y trabajos futuros.

II. Un esquema de metaheurísticas cooperativas

Para coordinar las diferentes metaheurísticas se pueden considerar varios esquemas. Un esquema interesante es el que se está utilizando en la Universidad de Granada, [3]. En este esquema cada metaheurística está representada por un agente y existe un agente coordinador que modifica sus comportamientos.

Pasemos a ver cómo podría comportarse este tipo de sistema cooperativo para la resolución de la misma instancia mostrada en la figura 1.

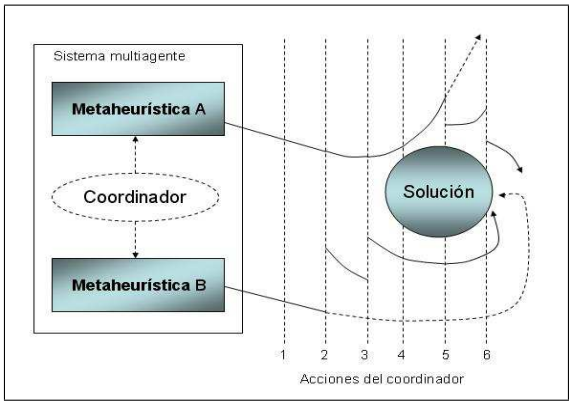

Fig. 2. Una posible ejecución de un sistema cooperativo

Ahora, en la figura 2, aparecen unas líneas verticales discontinuas referentes a las acciones que podría realizar el coordinador. En la primera línea la distancia a la solución es aproximadamente igual por parte de ambas metaheurísticas, así que el coordinador debería dejar proseguir. En la segunda, el coordinador podría detectar que la *metaheurística A* está más próxima a la solución que la B. Por esto sería interesante decidir cambiar la posición de la B en el espacio de búsqueda a una posición próxima a la de la A. Es decir, acercar la metaheurística B a la A. Por otro lado y por poner otro ejemplo, en la quinta

es la *metaheurística B* la que está cercana a la solución, por eso el coordinador cambiaría la posición de la *A* a una próxima a la de la *B*.

Observando en la figura 2 el resultado final podemos preguntar si se ha mejorado. Efectivamente, la mejora en este caso es considerable si recordamos lo que ocurría (representado en la figura con flechas discontinuas). Se consigue que la *metaheurística B* alcance la solución con mucho menos rodeo que cuando funcionaba en aislado (que se traduce en mucho menos tiempo de ejecución). De la misma manera la *metaheurística A* ha quedado mucho más próxima a la solución que al actuar sola. Es importante en este punto señalar que a lo que aspiramos es precisamente esto: óptimos locales de mayor calidad (como conseguiría la *metaheurística A*) y óptimos en menor tiempo (como conseguiría la *metaheurística B*).

Es interesante aclarar cómo se realizaría la comunicación de los agentes de este sistema. Se puede hacer por medio de un modelo de pizarra, en el cual los agentes toman y dejan la información necesaria para la cooperación en una estructura de datos llamada pizarra. No obstante, se ha adaptado este esquema para el sistema de metaheurísticas cooperativas. Ahora las metaheurísticas únicamente están capacitadas para dejar información en la pizarra y será el coordinador el que tome información de ella para posteriormente dar órdenes directamente a las metaheurísticas. Este modelo de pizarra adaptada es más genérico ya que permite tareas tales como que el coordinador modifique el comportamiento de la ejecución de los agentes.

Hemos visto la funcionalidad de este coordinador que podría recopilar información del rendimiento de cada uno de las metaheurísticas y enviar órdenes que afecten a sus comportamientos de búsqueda. La siguiente cuestión es cómo va a saber actuar este coordinador, es decir, de dónde viene su inteligencia.

III. Un coordinador de metaheurísticas

El mencionado coordinador va a estar integrado por un conjunto de reglas fuzzy. Decidimos usar un sistema de reglas fuzzy ya que nos permitirá representar los datos de una forma más cercana al modo de razonar humano lo que hará más comprensible el sistema. Además esto permitirá incorporar fácilmente al sistema conocimiento experto, en forma de reglas *ad-hoc* (definidas por el propio usuario). Los principales tipos de reglas fuzzy son Mamdani y TSK. Nos decantamos por utilizar reglas Mandani debido a la gran interpretabilidad que presentan de dichas reglas. En este punto ya se puede dilucidar que uno de las claves del sistema metaheurístico cooperativo coordinado que buscamos es su sencillez e interpretabilidad.

Las reglas fuzzy van a tener la inteligencia para la coordinación, ¿pero de dónde surge esta inteligencia?. Las reglas fuzzy van ser el resultado de un proceso de extracción inteligente del conocimiento [1], [2], el cual se muestra en la figura 3.

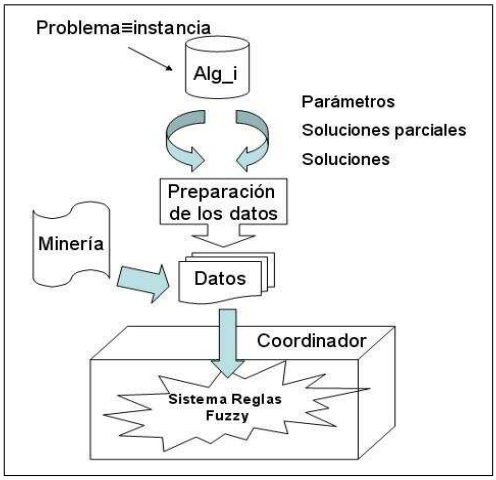

Fig. 3. Proceso de extracción del conocimiento

Este proceso comienza en la fase de preparación de los datos donde queremos obtener una base de datos que contenga la información útil a la cual podamos aplicar las técnicas de minería de datos. A continuación viene la fase de minería de datos en la que se espera obtener el modelo del coordinador del sistema metaheurístico utilizando técnicas de minería de datos. Y se finaliza con la fase de la evaluación del modelo donde se quiere comprobar la eficacia del conjunto de reglas fuzzy para modelar al coordinador.

IV. Proceso de extracción del conocimiento: Obteniendo un Prototipo

La definición del proceso de extracción del conocimiento, que puede verse en la figura 3 y que se explica con mayor detenimiento en [1], [2], comienza en la fase de preparación de los datos donde se pretende conseguir una base de datos que contenga información útil a la cual podamos aplicar las técnicas de minería de datos, obteniéndose esta información de la aplicación de distintas metaheurísticas sobre conjuntos de instancias de un problema. A continuación viene la fase de minería de datos, en la que se espera obtener un modelo del coordinador del sistema metaheurístico basado en reglas fuzzy utilizando técnicas de minería de datos. Y se finaliza con la fase de la evaluación del modelo donde lo que se quiere es comprobar la eficacia del conjunto de reglas fuzzy para modelar al coordinador.

A continuación mostraremos cómo se ha aplicado cada una de estas fases y los resultados que se han obtenido, para la construcción de un prototipo de modelado del coordinador, para el problema de la mochila, utilizando un algoritmo genético, búsqueda tabú, temple simulado y colonia de hormigas, como conjunto de metaheurísticas a cooperar.

A. Preparación de los datos

En esta fase es donde se escogió el problema y el conjunto de metaheurísticas y se resolvieron distintas instancias del problema, extrayéndose de las ejecuciones la base de datos.

El problema que se escogió es el problema de la mochila 0/1 por las siguientes razones:

- Queríamos un problema que fuera sencillo y estuviera bien estudiado de tal forma que se pudiera experimentar con el proceso de minería antes de aplicarlo a problemas más complejos como el de la p-mediana.
- De este problema además existen distintos tipos de instancias que van desde las muy sencillas de resolver a otras complejas. Esto es muy interesante para el objetivo de este trabajo, puesto que nos da la opción de encontrar fácilmente reglas que nos puedan indicar qué metaheurística va mejor con qué instancias y qué parámetros son más interesantes para cada metaheurística con cada tipo de instancia.

De este problema se usó una base de datos de instancias amablemente brindada por el Dr. Carlos Cruz, [3]. En ella se encuentran 4000 instancias del problema las cuales varían tanto en tamaño (número de objetos) como en la forma en la que han sido generadas. Además se conoce el óptimo de cada instancia. Lo que nos permitirá calcular la distancia a la que quedaron las metaheurísticas de la solución óptima.

Una vez seleccionado el problema se eligieron las metaheurísticas que se aplicarían para resolverlo. Para ello se realizó un estudio de las técnicas metaheurísticas más relevantes actualmente y en base a esta clasificación se decidió escoger cuatro metaheurísticas, de las cuales dos deberían ser basadas en trayectorias, y dos basadas en poblaciones, puesto que esta es una clasificación muy usada y queremos ver como el coordinador podría manipular cada tipo. Dentro de las metaheurísticas basadas en trayectorias, se escogieron el temple simulado y la búsqueda tabú. Y en cuanto a las metaheurísticas basadas en poblaciones se escogieron un algoritmo genético y una colonia de hormigas.

De estas cuatro metaheurísticas, tres se han utilizado dentro del proceso de extracción del conocimiento para completar lo que sería una iteración de este, incluyendo las fases de preparación de datos y minería. Y una cuarta se ha usado para ser añadida al sistema una vez se haya modelado este completamente. De tal forma que se simule lo que significaría la adición de una metaheurística tras haber evaluado este y comprobado que presenta ciertas carencias. Las tres metaheurísticas que se han seleccionado, de forma arbitraria, para la primera fase son la búsqueda tabú, el algoritmo genético y el temple simulado. Quedando por tanto la colonia de hormigas para ser añadida después.

Una vez elegido el problema y las metaheurísticas que lo resolverán pasamos a recopilar la información que se utilizará para obtener la definición del coordinador. De tal forma que se aplicaron las distintas metaheurísticas elegidas a la base de datos de instancias, extrayéndose de estas ejecuciones cierta información interesante, con la que se genera la estructura de un ejemplo, formado por un vector definido de acuerdo a una serie de características. En esta estructura se almacena todo tipo de información: características heterogéneas, tanto numéricas como nominales, pudiendo contener estas valores imperfectos (con incertidumbre e imprecisión) y además permite que el valor de alguna característica de un ejemplo concreto sea desconocido.

Finalmente para construir la base de datos se seleccionaron de cada instancia las ejecuciones de cada metaheurística que mejor resultado obtuvieron y se fusionaron en un vector, de tal manera que, como muestra la tabla I, un ejemplo está compuesto por los siguientes atributos:

- Una descripción de la instancia del problema que incluye: nombre de la instancia, peso máximo de la mochila, beneficio óptimo, número de objetos disponibles, tipo de la instacia y una etiqueta lingüística que indique la dificultad de la instancia.
- La configuración de parámetros de cada metaheurística que mejor resultado dio.
- Información obtenida durante las ejecuciones de cada metaheurística que incluye: beneficio de la solución inicial, dos soluciones intermedias y la final, así como la proporción de error con respecto al óptimo de la solución final.

TABLA I
UN TROZO DE BASE DE DATOS CONSTRUIDA

Campos con información de la instancia:					
FICHERO	**nobjetos**	**capacidad**	**bobtimo**	**TIPO**	**DIF**
[kp500 16 1]	(500)	(5430332)	(10457924)	[circle]	[dificil]
[kp500 16 2]	(500)	(5151503)	(10212388)	[circle]	[dificil]
[kp500 16 3]	(500)	(5058489)	(10081239)	[circle]	[dificil]
...	...	...	...	...	...
Campos con los parámetros de los algoritmos:					
nindiv	**generac**	**pcruce**	**pmutacion**	**FUN**	**pvecinos**
(700)	(5000)	(0.4)	(0.001)	[E]	(90.00)
(700)	(5000)	(0.4)	(0.001)	[E]	(90.00)
(700)	(5000)	(0.4)	(0.001)	[E]	(90.00)
...	...	...	...	...	...
itertemple	**itemp**	**ftemp**	**itertabu**	**tlt**	**cd**
(1799)	(1E+15)	(0.00)	(10000)	(100)	(2.0)
(1799)	(1E+15)	(0.00)	(10000)	(100)	(1.0)
(1799)	(1E+15)	(0.00)	(10000)	(1000)	(1.5)
...	...	...	...	...	...
Campos con las soluciones para el algoritmo genético:					
bsig	**bsi1g**	**bsi2g**	**pobtgenet**	**bobtgenet**	**dcgenet**
(8735546)	(9360220)	(9798914)	(5430291)	(10173741)	(284183)
(8504904)	(9735169)	(9840810)	(5151500)	(9954090)	(258298)
(8375944)	(9686237)	(9755172)	(5058444)	(9827887)	(253352)
...	...	...	...	...	...
Campos con las soluciones para el algoritmo temple simulado:					
bsit	**bsi1t**	**bsi2t**	**pobttemp**	**bobttemp**	**dctemp**
(5906547)	(9114006)	(9114006)	(5430322)	(9114006)	(1343918)
(3054866)	(8745338)	(8747108)	(5151488)	(8747108)	(1465280)
(6918986)	(8681853)	(8681853)	(5058448)	(8681853)	(1399386)
...	...	...	...	...	...
Campos con las soluciones para el algoritmo tabú:					
bsitabu	**bsi1tabu**	**bsi2tabu**	**pobttabu**	**bobttabu**	**dctabu**
(8509765)	(8705886)	(8713846)	(5430330)	(8779676)	(1678248)
(8218710)	(8484099)	(8402945)	(5151501)	(8546851)	(1665537)
(8162823)	(8314109)	(8400753)	(5058480)	(8433190)	(1648049)
...	...	...	...	...	...

Una vez obtenida la base de datos se aplicó un preproceso de tal forma que se seleccionaron aquellos atributos e instancias más relevantes para cada inferencia que se desee realizar. Para realizar este paso nosotros realizamos la selección tanto de un

modo ad hoc, utilizando nuestro conocimiento, como haciendo uso de técnicas como las que nos ofrece la herramienta Weka, [8].

Una vez aplicado este paso se obtiene la base de datos a la que se podrán aplicar técnicas de minería de datos.

B. Minería de datos

En esta fase se aplicó una técnica de minería de datos sobre la base de datos conseguida en la anterior fase, obteniéndose como resultado un conjunto de reglas fuzzy de bajo nivel que pretenden modelar el coordinador del sistema.

Para poder aplicar esta fase es necesario escoger una técnica de minería de datos. Existe una amplia variedad de técnicas de minería de datos, basadas en diferentes propuestas teóricas. Nosotros buscamos una técnica que sea interpretable, de tal manera que se puedan comprender fácilmente los modelos que devuelve, y que además permita tratar datos imperfectos.

Finalmente nos decantamos por árboles de decisión, puesto que es una de las técnicas más interpretables, y además es muy utilizada, por lo que se han hecho grandes esfuerzos por incorporar en ella el tratamiento de datos imperfectos. Dentro de los árboles de decisión existen diferentes alternativas disponibles, de las cuales hemos considerado CART, ID3, C4.5 o FID3.4. Las dos primeras (CART e ID3) fueron descartadas puesto que no incorporan tratamiento de datos imperfectos (imprecisos e inciertos). C4.5 aparece como respuesta a esto, ya que generaliza el algoritmo básico de construcción de árboles ID3 para tratar datos tanto nominales como continuos y que además permite el tratamiento de datos con un cierto tipo de imperfección. Entre estos tipos de imperfección sin embargo no se encuentran los conjuntos fuzzy (etiquetas lingüísticas). En este sentido, FID3.4 [5] mezcla la representación fuzzy y sus capacidades de razonamiento aproximado con la generación de árboles de decisión simbólicos uniendo las ventajas de ambos: manejo de datos imperfectos junto con el alto grado de popularidad, comprensibilidad y fácil aplicación de esta técnica. Siendo éstas las razones que nos han impulsado a escoger esta última técnica como la que usaremos en la fase de minería.

Una vez elegida la técnica de minería de datos la aplicamos a distintos conjuntos de la base de datos previamente seleccionados mediante la aplicación del preproceso. Teniendo como objetivos:

1. Realizar un estudio comparativo del comportamiento de las distintas metaheurísticas implementadas. De tal forma que podamos determinar:
   - Cómo se comporta cada metaheurística a lo largo del tiempo con respecto al resto de metaheurísticas. Es decir, si durante un intervalo de tiempo converge más rápido que el resto, si se queda estancada en un determinado momento, etc.
   - Cuál es el comportamiento de cada metaheurística con cada tipo de instancia. Es decir, si alguna de ellas se comporta mejor que el resto con algún tipo de instancia.
   - Una mezcla de las dos anteriores. Por ejemplo, si una metaheurística converge muy rápido durante un intervalo de tiempo con las instancias de un determinado tipo.

2. Realizar un estudio de los parámetros de cada metaheurística de tal forma que podamos observar:
   - Qué combinaciones de parámetros funcionan mejor.
   - Qué conjunto de parámetros es más adecuado para cada tipo de instancia.
   - Qué conjuntos de parámetros pueden sustituir a los que se consideran los mejores.

Como resultado de la aplicación de FID3.4 sobre los distintos subconjuntos de la base de datos se obtuvieron varios conjuntos de árboles como el que se muestra en la tabla II. En este árbol se pretendía inferir el número de individuos del algoritmo genético usando para ello la dificultad de la instancia que se resolvía, el error que se obtuvo con el algoritmo genético, la probabilidad de mutación y la probabilidad de cruce.

TABLA II
UN ÁRBOL DE DECISIÓN OBTENIDO CON FID3.4

	INFERIR NINDIVIDUOS	N inst	300	700
1	Totales	2000.00	413.00	1587.00
2	[dcgenetico=cero]	891.50	376.98	514.52
3	[dcgenetico=cero][pcruce=40]	655.83	341.22	314.61
4	[dcgenetico=cero][pcruce=60]	235.67	35.76	199.91
5	[dcgenetico=muy p]	774.62	30.77	743.85
6	[dcgenetico=acep]	282.94	5.25	277.69
7	[dcgenetico=regul]	50.94	0.00	50.94

Interpretando los árboles obtenidos en FID3.4 se pudo definir un extenso conjunto de reglas compuesto por 37 reglas fuzzy de bajo nivel. Para crear estas reglas previamente tuvieron que definirse distintos conjuntos fuzzy para los distintos valores que pueden tomar: las diferencias entre los beneficios obtenidos por cada metaheurística, el tiempo y los parámetros que puede tomar cada metaheurística. En la figura 4 se pueden ver, como ejemplo, los conjuntos definidos para la diferencia entre los beneficios obtenidos por cada metaheurística.

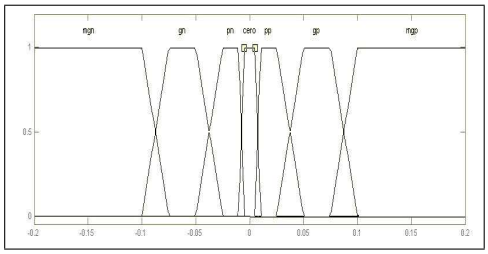

Fig. 4. Conjuntos fuzzy de diferencias de beneficios

Usando estos conjuntos fuzzy se crearon las reglas, que cabe destacar, que modelaron los siguientes com-

portamientos:

▪ Cuándo y de qué manera se puede cambiar la solución de una metaheurística por la solución de otra metaheurística que se está comportando mejor. Ejemplos de este tipo de reglas se pueden ver en la tabla III.

TABLA III
REGLAS OBTENIDAS PARA EL CAMBIO DE SOLUCIÓN

SI [Tiempo ES *Reinicio*1 Y Dificultad ES *Dificil* Y p_gen_tem ES (*gp* O *mgp*)] ENTONCES Reiniciar temple con solución de genético.
SI [Tiempo ES *Reinicio* Y Dificultad ¬ ES *muyFacil* Y p_gen_tab ES *mgp*] ENTONCES [Reiniciar con solución de Genético]
SI [Tiempo ES *Reinicio* Y Dificultad ES *muyFacil* Y p_gen_tab ¬ ES *mgp* Y (p_tem_tab ES *mgp* O p_tem_tab ES *gp*)] ENTONCES [Reiniciar con solución de Templado]
.....

▪ Con qué parámetros se debe iniciar cada metaheurística dependiendo del tipo de instancia que se desee resolver. Ejemplos de este tipo de reglas se pueden ver en la tabla IV.

TABLA IV
REGLAS OBTENIDAS PARA EL INICIO

SI [Tiempo ES *Inicio*] ENTONCES pcruce ES *b*
SI [Tiempo ES *Inicio* Y Dificultad ES *muyFacil*] ENTONCES nindividuos ES *b*
SI [Tiempo ES *Inicio*] ENTONCES fun ES *C* Y tmax ES *mb* Y tmin ES *b* Y pvecinos ES *a*
SI [Tiempo ES *Inicio* Y Dificultad ES *muyFacil*] ENTONCES [cd ES *b* Y tlt ES *b*]
.....

▪ Cuándo y cómo se pueden cambiar los parámetros de una metaheurística que no está obteniendo el comportamiento esperado. Ejemplos de este tipo de reglas se pueden ver en la tabla V.

TABLA V
REGLAS OBTENIDAS PARA EL CAMBIO DE PARÁMETROS

SI [p_gen_tab ES (*pn* O *gn* O *mgn*) Y Tiempo ES *cp*6 Y Dificultad ES *muyfacil*] ENTONCES pcruce ES *a*
SI [Tiempo ES *Segundocptrasreinicio* Y Dificultad ES *media* Y p_tem_tab ES (*pn* O *gn* O *mgn*)] ENTONCES fun ES *R* Y tmax ES *b* y tmin ES *a*
SI [Tiempo es *Segundocptrasreinicio* Y Dificultad ¬ ES *muyfacil* Y (p_gen_tab ES *mgp* O p_tem_tab ES *mgp* O p_tem_tab ES *gp*)] ENTONCES [cd ES *b* Y tlt ES *b*]
.....

▪ Cuándo y de qué forma se puede cambiar la solución de una metaheurística por la de otra que ha demostrado comportarse mejor durante un intervalo de tiempo.

▪ Qué regla debe seleccionarse en caso de que más de una se active.

C. Modelado del coordinador

Las reglas que hemos obtenido en el punto anterior son perfectamente válidas al venir directamente del conocimiento conseguido en la fase de minería de datos. No obstante, estas reglas tienen algunas carencias y pueden mermar la potencia del sistema metaheurístico cooperativo centralizado, puesto que son demasiado numerosas y poco abstractas como para servir de modelo del coordinador. Por lo que se decidió crear un conjunto de reglas más general obtenido a partir de las reglas anteriores y del comportamiento que se espera del coordinador.

De esta manera se obtuvieron tres reglas, aunque no se descarta crear alguna más.

▪ SI [($peso_1 * d_1$ O ... O $peso_n * d_n$) ES suficiente] ENTONCES cambiar la solución actual de la peor metaheurística.

▪ SI [tiempo $\in$ $[tiempo_1, tiempo_2]$] ENTONCES cambiar las soluciones actuales de las MetaX por la solución actual de MetaG.

▪ SI la Metaheurística *esMuchoPeorQueTodas* Y *esMomentoDeCambiarParametros* ENTONCES *cambiarParametros* de Metaheurística.

La primera regla intenta indicar cómo se puede cambiar la posición en el espacio de búsqueda de la metaheurística que está obteniendo un peor resultado por una posición más cercana a otra metaheurística con mejor comportamiento. De esta manera a cada metaheurística se le asigna un peso utilizando el conocimiento obtenido en la fase de minería de datos y este peso se multiplica por la diferencia entre la peor solución y la solución de la metaheurística. Si alguno de estos productos es suficientemente grande entonces se debe cambiar la solución de la metaheurística con peor resultado, por una cercana a la de la metaheurística o metaheurísticas que hicieron saltar la regla.

La segunda regla nos indica que si se observa en los datos que una metaheurística, llamémosla MetaG, obtiene durante un intervalo de tiempo unas soluciones con un beneficio mucho mayor que las del resto, entonces durante ese intervalo de tiempo se pueden cambiar las soluciones del resto de metaheurísticas por las suyas.

Por último la tercera regla muestra como modificar los parámetros de las distintas metaheurísticas. De esta forma si una metaheurística obtiene unas soluciones con un beneficio mucho menor que las del

resto y hace bastante tiempo desde que se le cambiaron los parámetros, entonces se le puede dar un nuevo conjunto de parámetros utilizando las reglas de bajo nivel que se definieron anteriormente.

D. Evaluación del modelo

En esta fase lo que se quiere es comprobar la eficacia del conjunto de reglas fuzzy para modelar al coordinador y que éste realice su trabajo de forma eficiente con respecto al coste computacional y a la bondad de las soluciones obtenidas.

Para medir el comportamiento del sistema, utilizaremos varias medidas como el error cuadrático medio y el error cuadrático relativo, utilizando como metodología para la realización de los experimentos los métodos de validación cruzada y bootstrap. Además, utilizaremos la información que nos proporcionará el sistema con respecto a las reglas que se han activado, qué algoritmos se han visto modificados en su ejecución, qué algoritmos parece que no tienen un comportamiento adecuado con respecto a los demás y a la solución que obtienen, etc.

Una vez ejecutadas todas las pruebas, calculados los métodos de evaluación y obtenida la información relevante de las ejecuciones, podremos resolver si el comportamiento es aceptable o si por el contrario debemos modificar (retroalimentación) el sistema, abordando de nuevo el proceso de extracción de conocimiento (eliminar/añadir características, eliminar/añadir la información correspondiente a un algoritmo, cambio de técnica de minería de datos, etc) con el objetivo de mejorar el comportamiento del coordinador utilizando esta información.

Como resultado de este proceso se puede determinar que es necesario:

- Eliminar una metaheurística en caso de que la información obtenida indique que no aporta nada al sistema.
- Añadir o eliminar características relativas a una metaheurística en caso de que su comportamiento haya empeorado con respecto a su comportamiento individual.
- Añadir o eliminar características relativas a un conjunto de instancias en caso de que una metaheurística haya empeorado su comportamiento con respecto al inicial.
- Y en otro caso, aplicar una nueva técnica de minería para obtener otro tipo de reglas.
- O añadir una metaheurística.

Una vez se haya hecho esto se puede volver a aplicar el proceso de extracción del conocimiento para mejorar el sistema.

En esta fase de evaluación, se ha planteado añadir un algoritmo metaheurístico nuevo al sistema una vez este se ha modelado completamente. De esta forma se añadirá al modelo ya obtenido el algoritmo metaheurístico colonia de hormigas (ACO) volviéndose a aplicar el proceso de extracción del conocimiento pero ahora con el objetivo de añadir una nueva metaheurística.

De igual manera que cuando se modelaba el sistema desde el principio, lo primero que se debe hacer es aplicar ACO a la base de datos de pruebas y recopilar los datos que se generen. Una vez se ha añadido esta información a la base de datos se puede aplicar un preproceso a esta y pasar a la fase de minería con el objetivo de realizar una comparativa entre los cuatro algoritmos y extraer los parámetros más interesantes para el algoritmo en cada momento. Así obtuvimos los parámetros a usar e incluimos en la jerarquía ACO. Con esta información se crearon de nuevo reglas de bajo nivel, que finalmente se incluyeron en el coordinador.

Podría pensarse que la agregación de una nueva metaheurística al sistema implica modificaciones en las reglas que modelan el coordinador. Afortunadamente se ha conseguido robustez en este proceso. Las reglas son lo suficientemente abstractas como para funcionar con el nuevo conocimiento introducido. Y solamente se debe modificar la regla que incluye pesos para que tenga en cuenta la colonia de hormigas y añadir reglas a un nivel más bajo para incluir el nuevo conocimiento en el coordinador.

V. Conclusiones y Trabajos Futuros

Durante el desarrollo de este artículo se ha expuesto la construcción de un prototipo para el modelado del coordinador de un sistema metaheurístico cooperativo centralizado. Para esto, se ha realizado un proceso de extracción del conocimiento, y el producto final del proceso ha sido un conjunto de reglas que modelan el coordinador. Todo este proceso ha sido aplicado a un problema concreto, el problema de la mochila.

El proceso de extracción de conocimiento ha comprendido:

- Preparación de los datos: Se ha realizado un análisis de la estructura de la base de datos requerida, estudiado las metaheurísticas que más no interesaban, aplicado estas metaheurísticas a instancias y extraido información interesante de estas ejecuciones.
- Minería de datos: En la fase de minería de datos se ha realizado un estudio de las técnicas de minería de datos que pueden ser más adecuadas, comparado el comportamiento de las metaheurísticas, realizado un estudio de los parámetros de estas metaheurísticas, enumerado conocimiento conseguido y producido reglas fuzzy.
- Modelado del Coordinador: En este punto se han adaptado las reglas que se consiguieron anteriormente con el fin de disponerse para la integración inmediata en el coordinador.

▪ Evaluación del modelo: Se ha planteado añadir un algoritmo al sistema una vez se ha terminado este y demuestrado la robustez del coordinador frente a la agregación o eliminación de metaheurísticas.

El sistema metaheurístico cooperativo obtenido podemos verlo en la figura 5.

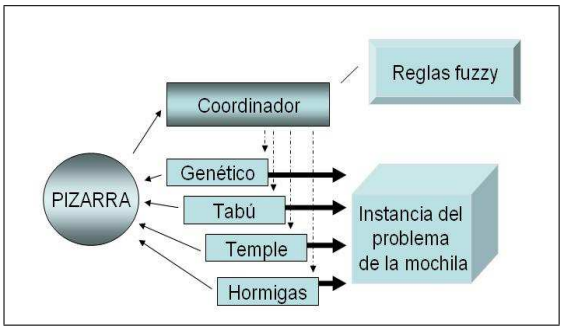

Fig. 5. Modelado del coordinador para este sistema

La información conseguida es valiosa en el uso tradicional de las metaheurísticas, ya que los estudios sobre la selección de metaheurísticas, comparativa de metaheurísticas frente al problema de la mochila y parámetros de cada metaheurística pueden ser muy interesantes para el lector. Pero además, más allá de este uso, el sistema de reglas obtenido proporciona inteligencia al coordinador para:

▪ Cuando un algoritmo esta más alejado de la solución que otros, cambiar su posición en el espacio de búsqueda por otra posición cercana a la que tenga otro algoritmo más eficiente.
▪ Cuando un algoritmo persiste en un mal comportamiento, cambiar sus parámetros de manera inteligente.

Respecto a los trabajos futuros, el inmediato es probar la descripción del coordinador del sistema metaheurístico centralizado, esto es, el conjunto de reglas fuzzy obtenidas, en una implementación de dicho coordinador. Está planeado que la implementación del coordinador la lleve a cabo el departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

En otra línea de avance se espera poder realizar el mismo proceso que se ha seguido para el problema de la p-mediana. Con estas instancias se seguirá el mismo proceso aquí expuesto para obtener la descripción de un agente coordinador de un sistema metaheurístico cooperativo centralizado para resolver el problema de la p-mediana.

Agradecimientos

Los autores agradecen al "Ministerio de Educación y Ciencia" y al "Fondo Europeo de Desarrollo Regional" (FEDER) por el soporte, bajo los proyectos TIN2005-08404-C04-02 y TIN2005-08404-C04-04, para el desarrollo de este trabajo.

Referencias

[1] J.M. Cadenas, M.C. Garrido, L.D. Hernández, E. Muñoz, *Towards a definition of a Data Mining process based on Fuzzy Sets for Cooperative Metaheuristic systems*, International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU2006, 2828–2835, París, 2006.

[2] J.M. Cadenas, R.A. Díaz-Valladares, M.C. Garrido, L.D. Hernández, E. Serrano, *Modelado del Coordinador de un sistema Meta-Heurístico Cooperativo mediante SoftComputing*, Campus Multidisciplinar en Percepción e Inteligencia, CMPI2006, 679–689, Albacete, 2006.

[3] C.A. Cruz, *Estratégias coordinadas paralelas basadas en soft-computing para la solución de problemas de optimización*, Tesis doctoral del Dpto. de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada, 2005.

[4] H. Guo, *A Bayesian Approach for Automatic Algorithm Selection*, IJCAI03 Workshop on AI and Autonomic Comp., 2003.

[5] C. Z. Janikow, *Fuzzy Decision Tree/Forest. FID3.4*, URL: http://www.cs.umsl.edu/∼janikow/fid.

[6] B. Melián, J.A. Moreno, J.M. Moreno, *Metaheurísticas: un visión global*, Revista Iberoamericana de Inteligencia Artificial 19:7–28, 2003.

[7] J. R. Rice, *The algorithm selection problem*, Advances in Computers 15:65–118, 1976.

[8] University of Waikato *Weka, Data Mining with Open Source Machine Learning Software in Java*, URL: http://www.cs.waikato.ac.nz/ml/weka/.