Artículo de Investigación. Revista Ciencia e Ingeniería. Vol. 38, No. 2 pp. 99-106, abril - julio, 2017
ISSN 1316-7081. ISSN Elect. 2244-8780Universidad de los Andes (ULA)

Resolución computacional de un problema de optimización combinatorio hibrido

Computational resolution of a hybrid combinatorial optimization problem

Aguilar, Jose
CEMISID,Escuela de Ingeniería de Sistemas, Facultad de Ingeniería.
Universidad de Los Andes, Mérida 5101, Venezuela.
Investigador Prometeo en la Universidad Técnica Particular de Loja, Ecuador
aguilar@ula.ve

Resumen

En este artículo definimos un problema de optimización combinatorio hibrido, y exploramos varias heurísticas computacionales de resolución del mismo. El problema de optimización combinatorio hibrido consiste en los problemas de la mochila y el viajero de comercio. Este problema hibrido tiene aplicaciones muy importantes, en particular, aquí lo exploramos en el ámbito de la robótica, para el caso en el que se requiere optimizar el comportamiento de los robots en la búsqueda de múltiples objetos en varios recorridos, según sus capacidades. Se requiere minimizar el número de recorridos y que cada recorrido sea lo más corto posible, pasando por los sitios que contienen los objetos a recoger en ese recorrido. Ese problema es muy importante en los supermercados automatizados, en particular para la gestión de los almacenes cada vez que llega un pedido de los clientes. En este artículo planteamos el problema, así como analizamos su resolución usando varias meta-heurísticas, en específico, el recocido simulado, los algoritmos genéticos, las redes neuronales aleatorias y los algoritmos tabú. Además, presentamos el ejemplo de aplicación en robótica y algunos resultados.

Palabras claves: Problemas de optimización combinatoria, algoritmos heurísticos, automatización de supermercado, ambientes inteligentes.

Abstract

In this article, we define a hybrid combinatorial optimization problem and explore various computational heuristic to solve it. The hybrid combinatorial optimization problem consists of the knapsack problem and the traveling salesman problem. This hybrid problem has very important applications, in particular, in this paper we explore the field of robotic, to optimize the behavior of robots in search of multiple objects on multiple tours, according to their abilities. In this case, it is required to minimize the number of routes, and each route must be as short as possible, through sites that contain objects to pick up on that tour. This problem is very important in automated supermarkets, particularly, to warehouse management when an order comes from customers. In this article we pose the problem, and analyze its resolution using various meta-heuristics, specifically, simulated annealing, genetic algorithms, artificial neural networks and the taboo random algorithms. In addition, we present an example of application in robotic, and some results.

Keywords: Combinatorial optimization problems, heuristic algorithms, supermarket automation, intelligent environments.

1 Introducción

La optimización combinatoria es un vasto campo de la matemática aplicada, que desde la perspectiva computacional busca el desarrollo de soluciones algorítmicas a problemas donde se requiere optimizar (maximizar o minimizar) una función objetivo.Algo relevante de estos problemas es que la palabra combinatoria se refiere al hecho que únicamente existe un número finito de soluciones factibles, pero además, que ese conjunto factible es de gran talla.

Esta disciplina tiene numerosas aplicaciones a nivel industria, social, en la administración de organizaciones, etc. Como ejemplos podemos mencionar, entre otros, el enrutamiento de mensajes en las redes de telecomunicación, la planificación de la producción industrial, la asignación de tareas a procesadores en un entorno distribuido, etc. Como se ve, hay una gran variedad de aplicaciones, que además, si se combinan, permiten extender sus posibles usos.

Ese es el caso en este trabajo, en el cual queremos resolver dos problemas de optimización combinatoria simultáneamente. El problema hibrido a estudiar combina el de la mochila y el del viajero de comercio, porque describe perfectamente el problema de recolección de productos en un almacén con el mínimo número de recorridos, optimizando cada recorrido en cuanto a hacerlo lo más corto posible según los objetos que deba recoger. Es el caso del problema de búsqueda óptima de una lista de objetos en un almacén. En nuestro caso particular, ese problema tiene su aplicación en el proceso de automatizar compras en un supermercado, usando robots para la recuperación de los pedidos de los usuarios en el almacén.

En general, la dificultad de los problemas de optimización combinatoria radica en buscar la solución óptima en un espacio de soluciones de elevada complejidad. En el caso de los métodos de búsqueda exactos, esta dificultad implica tiempos de cálculo inviables para hallar una solución óptima. Por otro lado, los métodos aproximados (por ejemplo, las meta-heurísticas), obtienen soluciones óptimas o cuasi-óptimas en tiempos aceptable (Puchingery col., 2010).

En el caso de un problema combinatorio hibrido como el que queremos resolver, cobra mayor importancia esos tiempos de resolución, por esa múltiple combinatoriedad que incorpora cada problema. Por esta razón, en este trabajo analizamos varias meta-heurísticas para la resolución del problema bajo estudio. En particular, analizaremos el comportamiento de los algoritmos genéticos, el recocido simulado, las redes neuronales aleatorias y el algoritmo tabú.

2 Marco Teórico

2.1 Problema de la Mochila

El problema de la mochila, abreviado como KP (por sus siglas en inglés *Knapsackproblem*), consiste en buscar la mejor solución a una situación análoga al llenar una mochila, con una capacidad dada para soportar un peso determinado, con todo o una parte de un conjunto de objetos, cada uno con un peso y valor específicos (si se quieren priorizar) (Puchingery col., 2010). Los objetos colocados en la mochila deben maximizar el valor total, sin exceder la capacidad máxima de la mochila.

Su definición formal es la siguiente: supongamos que tenemos n distintos tipos de ítems, que van del 1 al n. De cada tipo de ítem se tienen qi ítems disponibles, donde q_i es un entero positivo que cumple $1 \leq q_i \leq \infty$, para $\forall i=1,n$

Cada tipo de ítem i puede tener un beneficio asociado dado por v_i y un peso (o volumen) w_i. Usualmente se asume que los pesos no son negativos. Por otro lado, se tiene una mochila donde se pueden introducir los ítems, que soporta un peso máximo (o volumen máximo) W.

El problema consiste en meter en la mochila los ítems de tal forma que se maximice el valor de los ítems que contiene, siempre que no se supere el peso máximo que puede soportar la misma. La solución al problema vendrá dada por la secuencia de variables x_1, x_2, ..., x_n donde el valor de x_i indica cuantas copias se meterán en la mochila del tipo de ítem i.

$$\textbf{maximizar} \sum_{i=1}^{n} v_i x_i \qquad (1)$$

Tal que

$$\sum_{i=1}^{n} w_i x_i \leq W \text{ y } 1 \leq q_i \leq \infty,$$

Si $q_i = 1$ se dice que se trata del problema de la mochila 0-1. Además, si el conjunto de los pesos de los elementos es una secuencia súper-creciente, es decir, se verifica que:

$$w_i > \sum_{j=1}^{i-1} w_j \quad \forall i \qquad (2)$$

Entonces se dice que se trata de un problema de la mochila tramposa. Si en un problema de la mochila 0-1 los ítems están subdivididos en k clases, denotadas por N_i, y exactamente un ítem tienen que ser tomado de cada clase, entonces hablamos del problema de la mochila de múltiple elección. Si en un problema de la mochila 0-1 tenemos n ítems y m mochilas con capacidades W_i entonces tenemos el problema de la mochila múltiple.

2.2 Problema del Viajero de Comercio

El problema de asignación cuadrática (QAP, por sus siglas del Inglés *QuadraticAssignmentProblem*) es un problema de optimización combinatoria NP-duro que modela muchos problemas del mundo real (Dantzigy col., 2008, TSP 2015). En particular, en este trabajo estamos interesados en una versión de ese problema, el problema del Viajero de Comercio (TSP por sus siglas del inglés, *Travelling SalesmanProblem*).

El Problema del Viajero de Comercio responde a la siguiente pregunta: Dada una lista de ciudades y las distancias entre cada par de ellas, ¿cuál es la ruta más corta posible que visita cada ciudad exactamente una vez y regresa a la ciudad origen? En otras palabras, una persona debe visitar un conjunto de m ciudades, comenzando en una ciudad determinada y finalizando en la misma ciudad; luego de haber visitado todas ellas sólo una vez. Esto significa que nunca regresa a una ciudad ya visitada, excepto la primera.

Es uno de los problemas de optimización más estudiados y es usado como prueba para muchos métodos de optimización. Existe una gran cantidad de heurísticas y métodos exactos son conocidos.

El TSP puede ser modelado como un grafo con peso en los arcos no dirigido, de manera que las ciudades sean los vértices, los caminos son los arcos y las distancias entre dos ciudades son los pesos de los arcos. Esto es un problema de minimización que comienza y termina en un vértice especifico y se visita el resto de los vértices exactamente una vez.

En el 'TSP simétrico' la distancia entre un par de ciudades es la misma en cada dirección opuesta, formando un grafo no dirigido. En el 'TSP asimétrico' pueden no existir caminos en ambas direcciones o las distancias pueden ser diferentes, formando un grafo dirigido.

El TSP se puede formulado matemáticamente de la siguiente manera: Dada N ciudades, el viajero de comercio debe visitar cada ciudad una vez, teniendo en cuenta que el costo total del recorrido debe ser mínimo. Eso puede modelarse como un grafo G:

G=(N, A)
N={1, ...,n}: conjunto de n nodos (vértices)
A={a_{ij}}: matriz de adyacencia.
d_{ij} = { $\propto$Si $a_{ij} = 0$
LijSi $a_{ij} = 1$
L_{ij}: distancia entre las ciudades i y j.

Suponiendo que las ciudades son numeradas desde 1 hasta n, una solución al problema puede expresarse a través de una matriz de estado E:

e_{ij} = { 1 Si la ciudad j fue la iesima ciudad visitada
0 En otro caso

La matriz E permite definir un arreglo unidimensional V de dimensión n;

$v_j = i$ Si la ciudad i fue la j-esima ciudad visitada

La función a optimizar seria minimizar a F1, donde:

$$\mathbf{F1} = \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{j=1}^{n} \mathbf{L_{ik}e_{ij}e_{kj+1}} \quad (3)$$

las restricciones quedan definidas por F2

$$\mathbf{F2} = \sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{e_{ij}} - \mathbf{n}) + \sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{e_{ij}} - \mathbf{1}) + \sum_{j=1}^{n}\sum_{i=1}^{n}(\mathbf{e_{ij}} - \mathbf{n}) \quad (4)$$

Tal que debe ser 0. El primer componente de esa expresión indica que los recorridos deben ser iguales a n, el segundo que cada ciudad debe ser visitada una vez, y el tercero que cada vez se visita solo a una ciudad.

El TSP tiene diversas aplicaciones, tales como: la planificación, y en la fabricación de microchips. En muchas aplicaciones, restricciones adicionales como el límite de recurso o las ventanas de tiempo hacen el problema considerablemente más difícil.

2.3 Meta-heurísticas para la solución de problemas combinatoria

En los últimos años ha habido un crecimiento espectacular en el desarrollo de procedimientos heurísticos para resolver problemas combinatorios. El auge que experimentan los procedimientos heurísticos se debe a la necesidad de disponer de herramientas que permitan ofrecer soluciones rápidas. Es importante destacar el hecho de que los algoritmos heurísticos normalmente no garantizan la optimalidad de la solución encontrada.

En particular, en (Vélez y col., 2007) introducen la siguiente definición de las Meta heurísticas: "son una clase de métodos aproximados que están diseñados para resolver problemas de optimización combinatoria, en los que los heurísticos clásicos no son ni efectivos ni eficientes".

Existen diferentes formas de clasificar y describir las meta-heurísticas (Vélez y col., 2007). Dependiendo de las características que se seleccionen se pueden obtener diferentes taxonomías: basadas en la naturaleza y no basadas en la naturaleza, con memoria o sin ella, con una o varias estructuras de vecindario, basadas en trayectoria y basadas en población

En los últimos años, el interés en combinar meta-heurísticas (meta-heurísticas híbridas) ha aumentado considerablemente (Vélez y col., 2007). Nosotros, en este trabajo usaremos las siguientes:
Algoritmos Genéticos
Búsqueda Tabú
Redes Neuronales
Templado Simulado

En la siguiente sección no nos vamos a detener a explicar en detalles las técnicas, apenas daremos una introducción a las mismas por lo bien conocidas que son en

el área (para una extensa introducción ir a (Aguilar y col., 2001).

a) Algoritmos Genéticos

Son técnicas de búsqueda basadas en la selección natural y la genética (Aguilar y col., 2001). Se emula el proceso de información hereditaria en los organismos biológicos, la cual es pasada a través de los cromosomas que contienen la información de todos esos factores (los genes). Varios organismos se agrupan formando una población, y aquellos que mejor se adaptan son aquellos que tienen mayores probabilidades de sobrevivir y reproducirse. Algunos de los supervivientes son seleccionados para crear nuevos organismos. Además, los genes de un cromosoma pueden sufrir cambios produciendo mutaciones en los organismos.

En los algoritmos genéticos, los cromosomas representan a las soluciones, a los cuales se les aplica operaciones de reproducción, cruzamiento, mutación, entre otras, para generar nuevos organismos. La reproducción se guía a través de una función que mide el grado de adaptación del cromosoma, ya que el proceso de selección es dependiente de los valores que esa función le asigne a cada cromosoma: a mayor valor, mayor probabilidad de selección y supervivencia tendrá ese cromosoma. El cruce consiste en intercambian genes entre parejas. Las mutaciones consisten en alterar un bit de un cromosoma. El macro-algoritmo es:

Población inicial de individuos
Evaluación de los individuos
Reproducción (generación de nuevos individuos)
Reemplazo de individuos
Condición de finalización o regresa a paso 2

b) Búsqueda Tabú

Es un tipo de búsqueda por entornos. Es un proceso de búsqueda local para explorar el espacio de soluciones (Vélez y col., 2007). La búsqueda Tabú selecciona de una manera agresiva el mejor de los movimientos posibles en cada paso. Al contrario que en la búsqueda local, que siempre se mueve al mejor de su entorno y finaliza con la llegada a un óptimo local, la búsqueda Tabú permite moverse a una solución de su entorno o vecindad que no sea tan buena como la actual, de tal manera que pueda tener oportunidad de salir de óptimos locales, y continuar estratégicamente la búsqueda de soluciones aún mejores.

c) Redes Neuronales Artificiales

Emula el comportamiento del cerebro humano. Los modelos de redes neuronales intentan conseguir unos buenos resultados basándose en una interconexión de unos nodos computacionales llamados neuronas (emulando el modelo conexionista del cerebro). Las redes neuronales artificiales emulan al cerebro humano en dos aspectos (Aguilar y col., 2001, Aguilar, 1998):

- el conocimiento se adquiere mediante un proceso de aprendizaje, y
- la conexión inter-neuronal se utiliza para el almacenamiento del conocimiento.

d) Recocido simulado

Otra de las heurísticas muy conocidas y que tiene ganado un lugar importante en la resolución de problemas difíciles, es el Recocido Simulado.

El nombre e inspiración viene del proceso de recocido del acero y cerámicas, una técnica que consiste en calentar y luego enfriar lentamente el material para variar sus propiedades físicas. El calor causa que los átomos aumenten su energía, y que puedan así desplazarse de sus posiciones iniciales (un mínimo local de energía). El enfriamiento lento les da mayores probabilidades de recristalizar en configuraciones con menor energía que la inicial (mínimo global). El método es una adaptación del método de Montecarlo, y su macro-algoritmo es:

actual =solución_inicial[problema]
T= alta temperatura
repita hasta T= congelado
próximo= aleatoria selección desde actual de una solución vecina
E= costo[próximo]- costo[actual]
Si T= congelado y E> 0 entonces
regresar actual
de lo contrario
Si E>0 entonces
actual= próximo con probabilidad e E/ T
Si E< 0 entonces
actual= próximo
descender T si ya han sido aceptados un número dado de movimientos para ese T

Definición del problema de optimización combinatorio hibrido

Estado inicial: Posición actual y lista de productos solicitados.

Objetivo: Encontrar el mínimo número de recorridos, y en cada uno el mínimo camino a recorrer, para recolectar todos los productos de una lista dada, sin la necesidad de pasar más de una vez por el mismo lugar en un recorrido dado.

Espacio de estados (caminos): Recorrido por los diferentes sitios donde están los objetos.

Función de costo: Capacidad del agente de búsqueda y suma de distancias en cada recorrido.

Algoritmo de búsqueda a utilizar: Resolución de los problemas de la mochila y del viajero.

Descripción de la solución: La lista de productos es depurada mediante el algoritmo del problema de la mochila, de manera que tendremos pequeñas listas que cumplan con la cantidad máxima posible que puede recolectar el agente en un viaje. Después, cada recorrido será determinado mediante la aplicación del problema del viajero.

Salida: Todos los viajes (trayectoria óptima) necesarios para la recolección de todos los productos.

El macroalgoritmo es:

1 MiniLista: AplicarMochila(Lista)
% devuelve las minilistas a recoger en
% cada recorrido
2 Mientras (MiniLista No = Vacía){
3 AplicarViajero(MiniLista)
% devuelve Recorrido óptimo para esa minilista

3 Sacar actual MiniLista del conjunto de MiniListas

En el paso uno se define los recorridos a realizar (se distribuye la lista de productos a recoger en cada recorrido). En ese caso, se resuelve el problema de la mochila usando la ecuación 1. Después, en los pasos 2 y 3 se optimizan todos los recorridos para recoger los elementos identificados en la lista, resolviendo en este caso el problema del viajero de comercio usando las ecuaciones 3 y 4. Serán tantos recorridos como los definidos en el paso 1 (mini-listas definidas).

4 Casos de estudio

4.1 Benchmarks con diferentes heurísticas

En esta primera parte, analizamos el comportamiento de varias metas-heurísticas y la combinación de ellas, en varios grafos de pruebas. Aquí solo mostramos los resultados de las mejores combinaciones, o de las metas-heurísticas que individualmente dieron los mejores resultados. En el caso de las redes neuronales artificiales, usamos el modelo de Hopfield, por su capacidad de alcanzar estados estacionarios de baja energía (buenos óptimos locales).

En el primer caso exploramos la solución de varias meta-heurísticas partiendo de grafos que han sido usados como benchmarks para probar el problema TSP tomados de (TSP, 2015). A esos grafos, les hemos añadido la necesidad de recorridos parciales, según lo que indiquen las minilistas de nuestro algoritmo principal (paso 1).

Tabla 1.Costo de las mejores soluciones encontradas por cada meta-heurísticas para el primer grupo de benchmarks

Benchmark	AG	AG+RS	AG+MH	AG+AT	AT	MH
longley.tsp	10	12	10,1.	10,5	10.5	10.4
kleinlml.tsp	24,3	24,5	25	26	24,3	25,2
grunsur.tsp	34,2	35	35,3	34,2	35,6	34,5
pears.tsp	1003	103,4	104	105	104,3	105,3

En el segundo caso (tabla 2), consideramos benchamark que han sido definidos para probar el problema de la mochila. Esos benchmark lo que nos dan es que debemos recoger, pero no donde están (la lista de los objetos a recolectar, con la capacidad de la mochila). Los primeros benchmark son MPK (*Multidimensional KnapsackProblem* por sus siglas en inglés), el problema de la mochila multidimensional, el cual es una instancia de la librería de benchmark desarrollada por Chu and Beasley(Puchinger, 2010), con n=250 objetos y m=2 {5, 10} restricciones, y los otros benchmark son P06, que es el caso de un conjunto de 10 ítems y 2 morrales de capacidad de 103 y 156, respectivamente. Para ello, lo hemos combinado con algunos de los grafos anteriores (tabla 1), que nos dan ubicación.

Tabla 2.Costo de las mejores soluciones encontradas por cada meta-heurísticas para el primer grupo de benchmarks

benchmark	AG	AG+RS	AG+MH	AG+AT	AT	MH
MKP+ longley.tsp	13.4	13.2	13.3	12.4	11.1	11
MKP+ grunsur.tsp	44.5	40.1	38.1	40.1	39.9	39
P06 + longley.tsp	13.6	13.8	14.4	13.4	13.1	13.1
P06+grunsur.tsp	45	43.2	41.1	43.1	399	40

En la tabla 1 vemos que los AG dan los mejores resultados. La razón es que allí el problema de optimización combinatoria no se presenta en la primera fase, sino para buscar los recorridos más cortos (el problema de la mochila es muy simple), lo que implica que la complejidad de la segunda parte es alta, por lo que con un buen algoritmo de búsqueda global se resuelve rápidamente (los tiempos de ejecución son cortos para todos). Al final, nos quedábamos con la mejor solución hallada para ese momento. Eso nos dice que los algoritmos de búsqueda local son muy lentos en sus procesos de converger.

En la tabla 2 el mejor es la combinación con RS, ya que esa combinación perfectamente combina la búsqueda local y global. Allí también vemos que los algoritmos de búsqueda local, en generar, son bastante buenos. Eso es derivado a que en este caso el primer paso del algoritmo resuelve la complejidad del problema (reduce el espacio de soluciones) lo que permite ahora que una buena búsqueda local termine optimizando el problema.

4.1 Caso de estudio en robótica

El caso de estudio definido fue la recolección de un listado de productos solicitados por un cliente de un supermercado, en el almacén de dicho supermercado. El robot debe deducir cuantos recorridos debe hacer, y los mejores recorridos en cada caso. El robot recibe una lista con los siguientes productos y sus nodos (posición) asociadas:

*Galleta Soda A*Coca Cola A
*Crema dental C*Jamón serrano E

Donde el primer término se refiere al producto, y el segundo su ubicación en el almacén (ver figura 1).

Si todos estos productos no superan la capacidad máxima de la mochila, se realiza un solo recorrido. De lo contrario, el robot determina el número de recorridos y lo que debe recoger en cada uno (las mini-listas) (paso 1 de nuestro macro-algoritmo).

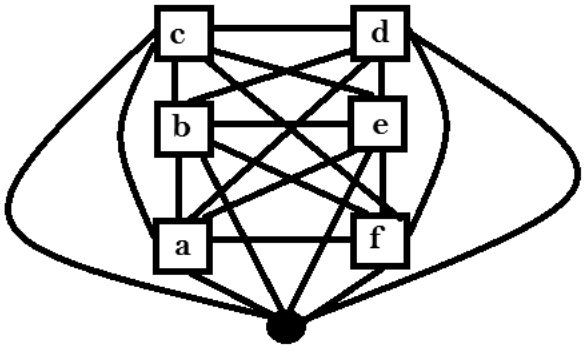

Fig 1. Almacén

Después, él determina los caminos óptimos de cada recorrido en el 3 paso, según los productos que deba recoger (contenidos en la respectiva mini-lista, ver figura 2):

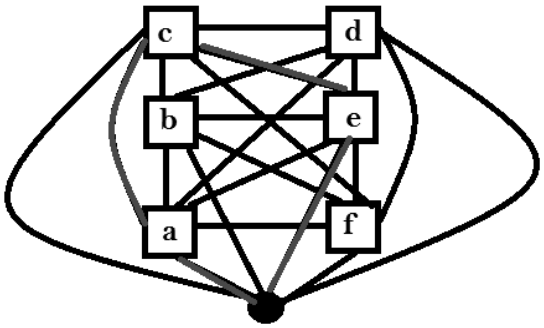

Fig 2. Camino óptimo para uno de los recorridos

Vemos que la instanciación del problema híbrido es mucho menos compleja en este caso que para el caso de los benchmarks de TSP, ya que la configuración de un almacén es mucho más sencilla (su grafo es menos complejo, no es un hiper-grafo, como el caso de los benchmarks de TSP).

La necesidad de cálculo que requiere el robot podría ser una limitante, por eso es preferible un algoritmo con poca necesidad de cómputo y sin requerimientos de mucha memoria. En ese caos, un AG simplificado, que dé una solución sub-optima en un tiempo de ejecución corto, eventualmente sería una buena solución (por ejemplo, se acotaría el número de generaciones).

5 Conclusiones

En este trabajo hemos analizado un problema de optimización hibrido, donde el orden de complejidad aumenta debido a la dimensionalidad de los problemas bajo estudio. Hemos visto que para los casos de una instanciación de gran dimensión, se requiere una buena combinación de metaheurísticas que hagan una búsqueda local y global eficientemente, para poder tener resultados aceptables en tiempos de ejecución cortos. Si se logra acotar el problema de dimensionalidad de cada problema, incluso los tiempos pueden mejorar contundentemente.

Por otro lado, el motivo de estudiar problemas de optimización combinatorio hibrido, es porque sus aplicaciones son muy importante en problemas reales. Por ejemplo, en entornos de automatización, como el ejemplo estudiando (búsqueda automática de productos en un almacén). Lo relevante de esta aplicación, es que las dimensiones del problema hibrido es mucho más pequeño, por lo que se pueden obtener resultados bastante eficientes (sub-óptimos) en tiempos de ejecución cortos. Para ello, con técnicas que limiten el uso de memoria y procesamiento sería suficiente

Próximos trabajos exploraran otros problemas de optimización híbridos de aplicación práctica, para determinar si esa regla comportamental de las metas-heurísticas se mantienen, en particular, las necesidades de combinar eficientemente los procesos de búsqueda local y global. Además,

para el mismo problema, se estudiaran otras técnicas de optimización bio-inspiradas como las propuestas en (Aguilar y col., 1998, Aguilar, 2001, Aguilar y col., 2004).

6 Agradecimientos

Dr. Aguilar actualmente es Investigador Prometeo en la Universidad Técnica Particular de Loja, Loja, Ecuador.

Referencias

Puchinger J, Raidl G, Pferschy U, 2010, The Multidimensional Knapsack Problem: Structure and Algorithms, Informs Journal on Computing, Vol. 22, pp. 250-265.

Dantzig G, Fulkerson D, Johns S, 2008, Solution of a Large-Scale Traveling-Salesman Problem, In 50 Years of Integer Programming, M. Jünger y col. eds., pp. 1958-1983.

TSP, Disponible en: http://tspintl.com/products /tsp/benchmarks/index.htm

Vélez M, Montoya J, 2007, Metaheurísticos: una alternativa para la solución de problemas combinatorios en administración de operaciones. Revista EIA, Vol. 8, pp. 99-115.

Aguilar J, Rivas F, (Eds.), 2001, Introducción a las Técnicas de Computación Inteligente, Meritec, Mérida.

Aguilar J, 1998, Definition of an Energy Function for the Random Neural to solve Optimization Problems, Neural Networks, Vol. 11, pp. 731-738.

Aguilar J, Hidrobo F, 1998, Toward a Parallel Genetic Algorithm Approach Based on Collective Intelligence for Combinatorial Optimization Problems, *Proceeding of the IEEE International Conference on Evolutionary Computation*. IEEE Neural Network Council, pp. 715-720, Alaska, USA.

Aguilar J. 2001, A General Ant Colony Model to solve Combinatorial Optimization Problems. Revista Colombiana de Computación. Vol. 2, pp. 7-18.

Aguilar J, Velázquez L, Pool M, 2004, The Combinatorial Ant System, *Applied Artificial Intelligence*, Vol. 18, pp. 427-446.

Recibido:*20 de enero de 2016*

Aceptado:*21 de octubre de 2016*

Aguilar, José: *obtuvo un Doctorado en la Universidad ReneDescartes-Paris-France y un Postdoctorado en la Universidad de Houston. Profesor Titular del Departamento de Computación de la Universidad delos Andes (ULA), miembro de Número de la Academia de Mérida y del Comité Técnico Internacional de la IEEE en Redes Neuronales. Ha publicado más de 500 artículos científicos en revistas científicas, libros y actas de congresos internacionales.*