5th International Conference on Industrial Engineering and Industrial Management
XV Congreso de Ingeniería de Organización
Cartagena, 7 a 9 de Septiembre de 2011

Estado del arte de algoritmos basados en colonias de hormigas para la resolución del problema VRP

David de la Fuente[1], Jesús Lozano[1], Eva Ochoa de Olano[1], Magín Díaz[1]

[1] Dpto. de Administración de Empresas de la Universidad de Oviedo. Escuela Politécnica de Ingeniería de Gijón. Campus de Viesques, s/n, 33204 Gijón. david@uniovi.es; lozano@uniovi.es; ochoadeolano.eva@gmail.com; magin_dr@hotmail.com.

Palabras clave: VRP, *Ant Colony Optimization, swarm intelligence*.

1. Introducción

El objetivo de este documento es el de realizar un estudio del estado del arte sobre la metodología actualmente aplicada a problemas de optimización combinatoria para la resolución de un problema real y concreto. El problema en cuestión es el de una empresa de logística que se dedica a la distribución de vehículos a concesionarios mediante transporte por carretera, empleando para ello camiones con remolques de capacidad variable. Este problema se trata en realidad de un subproblema de un Sistema de Transporte Intermodal de mayor envergadura*.

En una primera aproximación se podría clasificar el caso de estudio como un problema perteneciente a la bien conocida clase de problemas de enrutamiento de vehículos (VRP, *Vehicle Routing Problem*), al cual, posteriormente, se le irían añadiendo objetivos adicionales a conseguir para así ir incrementando la dificultad del planteamiento de forma progresiva.

El problema VRP más básico supone la existencia de un depósito central que cuenta con una flota de vehículos homogénea y debe atender a un conjunto de clientes geográficamente dispersos. El objetivo es entregar bienes a este conjunto de clientes con demandas conocidas, al mínimo coste, encontrando las rutas óptimas que se inician y terminan en el depósito. Cada cliente es servido una única vez y los vehículos de transporte llevarán la carga sin exceder su capacidad máxima disponible. Las características de los clientes, depósitos y vehículos, así como las diferentes restricciones operativas sobre las rutas, dan lugar a diferentes variantes del problema. A continuación se citan algunas de las más conocidas:

- Problema con restricción de capacidad (Capacitated VRP – CVRP)

* Este trabajo se deriva de la participación de sus autores en un proyecto de investigación financiado por el Ministerio de Fomento con referencia MFOM-08-E12/08, titulado “Análisis, desarrollo y evaluación de sistemas inteligentes de transporte de mercancías en un entorno intermodal”.

- Problema con ventanas de tiempo (VRP with Time Windows – VRPTW)

- Problema con múltiples depósitos (Multiple Depot VRP – MDVRP)

- Problema de entregas divididas con diferentes vehículos (Split Delivery VRP – SDVRP)

Además de estas variantes existen muchas más, e incluso pueden formularse modelos en los que se combinen varias de ellas.

El VRP es un problema de optimización combinatoria en el cual el número de soluciones factibles aumenta de forma exponencial con el número de clientes a visitar y se trata de una generalización del también conocido problemas TSP (*Travelling Salesman Problem*). El TSP es un problema NP-completo ampliamente estudiado debido a que es fácilmente descriptible pero difícil de resolver. Además, sirve de formulación para un gran número de casos prácticos del mundo real. Un problema NP es aquel para el que existe un algoritmo que verifica en tiempo polinomial para cada instancia si la respuesta "sí" es correcta. Se puede decir que los problemas NP-completo son los problemas más difíciles de la clase NP, por lo que existe una gran motivación para el estudio y obtención de heurísticas para su resolución.

El origen del problema VRP fue introducido por Dantzing y Ramser (1959), quienes describieron una aplicación real acerca de la entrega de gasolina a las estaciones de servicio y propusieron la formulación matemática a este problema. Cinco años después, Clarke y Wright (1964) propusieron el primer algoritmo que resultó efectivo para resolverlo, dando comienzo así a la amplia investigación en el área del enrutamiento de vehículos. Esta actividad es frecuente y costosa para muchas empresas, por lo que pequeñas mejoras en su eficiencia pueden provocar grandes reducciones de costes.

2. Planteamiento del problema

La empresa de logística divide el mapa de la geografía española en zonas de operación, cada una de las cuales contiene uno o más depósitos a los cuales llegan los pedidos y de los cuales se recogen los automóviles a distribuir. La gestión de los camiones de cada zona es llevada a cabo por un operador de tráfico, cuya función es la de asignar cargas a los camiones de forma que éstos cumplan una serie de restricciones y objetivos perseguidos y, posteriormente, despacharlos hacia su destino. Las restricciones a seguir son relativas a la capacidad (el camión no podrá transportar más carga de la que su capacidad máxima le permita) y a ventanas de tiempo (los pedidos no pueden ser entregados más tarde de la fecha indicada). En cuanto a los objetivos a conseguir se pretende, por un lado, maximizar el coeficiente de carga de cada uno de los camiones. Por otro lado se desea minimizar la cantidad de kilómetros que el camión viaja en vacío o con un factor de carga considerado demasiado pequeño. Por último es también interesante evitar la dispersión geográfica, es decir, intentar que las ciudades de destino se encuentren relativamente próximas al emplazamiento en el cual se cargue el camión.

Los camiones estarán disponibles para servir nuevos pedidos una vez que hayan finalizado su ruta actual y hayan entregado las cargas correspondientes. Puesto que se pretende evitar que los camiones se encuentren ociosos, es deseable que cada uno de ellos finalice su ruta en o cerca de lo que se denomina un punto de recarga óptimo (PRO). Un PRO se trata de un depósito en el cual, por experiencia previa y datos históricos, se sabe que llegará un gran

número de pedidos en instantes posteriores. De esta forma se evita tanto que un camión se encuentre estacionado en un almacén al que no llegan pedidos como que realice viajes con un bajo factor de carga. Esta forma de operar pone de manifiesto el hecho de que los camiones no realizan una distribución centralizada en torno a un único almacén, sino que pueden circular por todo el mapa saltando de zona en zona de operación. En otras palabras, no se impone la restricción de que un camión tenga que volver a su depósito de salida tras finalizar su reparto.

3. Búsqueda de soluciones: Métodos de Colonias de Hormigas

Una de las áreas de estudio más importante y prometedora de los últimos años es la investigación de algoritmos basados en comportamientos observados en la Naturaleza, los cuales permiten la obtención de métodos heurísticos no deterministas para la resolución de problemas de optimización combinatoria *NP-hard.* La Naturaleza presenta dos grandes mecanismos: la selección, que premia a los individuos más fuertes y penaliza a los más débiles; y la mutación, la cual introduce elementos aleatorios y permite el nacimiento de nuevos individuos. Aplicado a la obtención de heurísticas, la selección implementa la optimización y la mutación permite la realización de una búsqueda no determinista. Por otro lado, las heurísticas provenientes de la observación de la Naturaleza son adaptativas, ya que emplean técnicas de retroalimentación de información para así poder modificar los parámetros que describen el funcionamiento del modelo.

En los últimos años se ha producido una fuerte tendencia a la resolución de problemas de optimización combinatoria mediante la utilización de algoritmos ACO (*Ant Colony Optimization*). Estos algoritmos pertenecen al campo de la *swarm intelligence* o inteligencia de enjambre, y están compuestos por individuos simples que cooperan de forma auto-organizada, es decir, sin ninguna forma de control central sobre los miembros del enjambre. ACO es una técnica de optimización de propósito general basada en el comportamiento de colonias de hormigas reales, concretamente en las feromonas que depositan entre la comida y el nido para marcar de esta forma el mejor camino encontrado, Beckers et al. (1992) y Goss et al. (1989). La cantidad de feromonas en un camino aumenta cada vez que una hormiga lo atraviesa. A medida que esta cantidad incrementa, la probabilidad de que una hormiga siga ese camino también lo hace, por lo que la cantidad de feromonas en el camino más corto será mayor después de un determinado tiempo y, como consecuencia, un mayor número de hormigas tenderán a seleccionar dicho camino. Sin embargo, la decisión de seguir un camino o no nunca es determinista, por lo que se permite la continua exploración de rutas alternativas. Estos algoritmos utilizan también un procedimiento de evaporación que reduce la cantidad de feromona a lo largo del tiempo, poniendo así más énfasis en nuevas direcciones de búsqueda para evitar quedarse estancado por decisiones pasadas.

En la mayoría de estos algoritmos cada hormiga construye una solución según una regla de decisión basada en dos parámetros: valores de las feromonas locales (cómo de bueno era el movimiento en el pasado) e información heurística local (basada en una información a priori como por ejemplo la distancia del movimiento). Por tanto, uno de los aspectos más importantes en el estudio de algoritmos heurísticos es el balance entre intensificación y diversificación. Demasiado énfasis en la primera puede producir que los agentes converjan en un óptimo local y demasiado énfasis en la segunda puede causar un estado inestable. Sin embargo, estos dos factores son esenciales, ya que es necesario acelerar la convergencia y utilizar la diversificación para encontrar mejores soluciones.

3.1. Orígenes de los algoritmos ACO

El primer algoritmo ACO, propuesto por Colorni, Dorigo y Maniezzo (1991-1992), recibe el nombre de *Ant System* (AS). Sus autores definen el término "hormiga" como un agente simple local que interactúa con el resto de agentes y que presenta las siguientes características:

- Libera una sustancia denominada *trail* o rastro a lo largo del camino al viajar desde una ciudad a otra (las feromonas en las hormigas reales).
- Elige la próxima ciudad a visitar con una probabilidad que es función de la distancia a la ciudad (su inversa recibe el nombre de "visibilidad") y de la cantidad de rastro presente en el camino que las conecta.
- Realiza un recorrido legal, ya que las transiciones hacia ciudades ya visitadas son inhibidas hasta que la ruta sea terminada, empleando para ello una lista tabú.

En el algoritmo AS, cada hormiga genera una ruta completa seleccionando las ciudades de acuerdo a una regla probabilística de transición entre estados (las hormigas prefieren moverse hacia ciudades más cercanas y con un alto contenido de feromonas en su camino). Esta regla de transición presenta unos parámetros configurables que determinan la importancia relativa de la feromona (rastro) frente a la longitud del trayecto (visibilidad). Posteriormente se aplica una regla de actualización global de feromonas: se evapora una fracción de la feromona de los caminos no recorridos y cada hormiga deposita una cierta cantidad de la misma en los caminos pertenecientes a su ruta, en proporción a lo corta que ésta resultara. Finalmente se itera el mismo proceso.

Las diferentes formas posibles de computar la variación de feromonas a lo largo del tiempo y la selección del instante de actualización del rastro producen diferentes instancias de este algoritmo basado en hormigas: *Ant-density*, *Ant-quantity* y *Ant-cycle*. Una explicación más amplia de estas variaciones del algoritmo original, así como un mayor número de resultados computacionales obtenidos tras su aplicación para la resolución del bien conocido problema del TSP, pueden ser hallados en un artículo escrito por los mismos autores, Colorni et al. (1991). Posteriormente, Bullnheimer, Hartl y Strauss (1997) aplican el algoritmo AS a la resolución del problema VRP en su forma más básica, es decir, con restricciones de capacidad y distancias, un único almacén central y vehículos homogéneos. Para ello proponen un algoritmo AS híbrido, combinado el método original con una heurística 2-opt. Esta heurística consiste en que una ruta es mejorada borrando dos arcos, es decir, invirtiendo uno de los caminos resultantes y luego reconectándolos hasta que no pueda obtenerse ninguna mejora adicional.

Dos años más tarde (1999), los mismos autores presentan una mejora para este algoritmo híbrido, la cual consiste en la introducción en la fase inicial de una lista de candidatos para la selección de los clientes más prometedores. Esta lista de candidatos es obtenida ordenando el conjunto de las localizaciones de menor a mayor distancia. Dorigo y Gambardella (1997) desarrollaron, basándose en AS y con el objetivo de mejorar su eficiencia, el algoritmo ACS (*Ant Colony System*), cuya estructura ya presenta un cambio más sustancial con respecto al original. ACS difiere de AS en tres aspectos principales: i) la regla de transición entre estados proporciona una forma directa de balanceo entre exploración de nuevos caminos y explotación del conocimiento acumulado a priori; ii) la regla de actualización global es únicamente aplicada a los caminos pertenecientes a la mejor ruta y iii) mientras las hormigas construyen una solución se aplica una regla de actualización local de feromona. El algoritmo

comienza con el posicionamiento de *m* hormigas en *n* ciudades elegidas de acuerdo a alguna regla de inicialización (por ejemplo de forma aleatoria). Cada hormiga construye una ruta aplicando la regla de transició entre estados. A medida que construye su ruta, la hormiga actualiza la cantidad de feromona en los caminos recorridos aplicando la regla de actualización local. Una vez que todas las hormigas han terminado su ruta, la cantidad de feromona en los caminos es modificada de nuevo aplicando la regla de actualización global.Stüzle y Hoos (2000) propusieron el algoritmo MMAS (*MAX-MIN Ant System*) como una mejora del AS inicial. Sus características principales son que únicamente la mejor hormiga actualiza el rastro de feromonas y que el valor de la feromona está acotado, tanto superior como inferiormente. Estas cotas son típicamente obtenidas de forma empírica y ajustadas según el problema específico considerado por Socha et al. (2002).

3.2. Algoritmos ACO en la actualidad

Recientemente se han propuesto nuevos algoritmos basados en colonias de hormigas con el objetivo de obtener mejores resultados y rendimiento en la búsqueda de soluciones de problemas de optimización combinatoria más complejos. A pesar de que los algoritmos ACS y MMAS han demostrado tener un buen rendimiento, pueden presentar el fenómeno de quedarse estancados en un mínimo local como resultado de la acumulación de feromonas. Los algoritmos genéticos (GA, *Genetic Algorithms*), inicialmente propuestos por el profesor John Holland (1992), están basados en la idea de la evolución Darwiniana y son una poderosa herramienta para la resolución de problemas de optimización combinatoria. Muchos algoritmos ACO nuevos han sido propuestos mediante la introducción de los operadores genéticos (cruce, mutación y selección) en los métodos tradicionales, Kaveh et al. (2008) y Lee et al. (2008), para solventar los inconvenientes anteriormente mencionados, sin embargo, presentan una elevada complejidad y suponen un alto tiempo de computación. Por este motivo algunos autores han decidido incluir en los algoritmos ACO únicamente alguno de los operadores. Es el caso de la reciente propuesta del algoritmo MACO (*Mutated Ant Colony Optimization*), Zhao et al. (2010), en el cual se añade únicamente el operador de mutación. Esta mutación es utilizada para modificar de forma aleatoria uno o más elementos de la mejor solución local después de cada iteración. Si la solución mutada es mejor que la solución original, la primera sustituye a la segunda. En caso contrario, la mejor solución local permanece inalterada. De esta forma se mejora el comportamiento de la búsqueda local, se expande la diversidad de soluciones y se evita la convergencia prematura. La mutación se aplica al algoritmo MMAS (*mutated MMAS* o M-MMAS) y al algoritmo ACS (*mutated ACS* o M-ACS).

Otro algoritmo propuesto para solventar el enrutamiento de vehículos es EA (*Evolutionary Ant Algorithm*), propuesto por Tsai et al. (2002), el cual introduce también mecanismos genéticos (operadores de cruce y mutación) para evitar los mínimos locales y un método llamado NN (*Nearest Neighbour*) para obtener una solución de forma más rápida. La heurística NN es un método de búsqueda local que consiste en que el viajante comienza en una ciudad y posteriormente visita la ciudad más cercana a la inicial. Seguidamente vuelve a visitar la ciudad más cercana a la última en la que se encontraba y así hasta que todos los nodos sean visitados y el viajante vuelva a la ciudad de origen. La ruta creada de esta forma sirve a modo de ruta inicial que posteriormente será mejorada por el algoritmo EA.HK-ACO, Li et al. (2008), es otro algoritmo mejorado de optimización con colonias de hormigas que emplea límites inferiores de Held-Karp para estimar la longitud de la ruta óptima. Esta medida, relativamente rápida y fácil de computar, resulta muy práctica a la hora de evaluar soluciones cercanas a la óptima en grandes problemas donde el verdadero óptimo es desconocido.

PDACO (*Population Declining ACO*), Wu et al. (2008), es una propuesta de algoritmo que pretende aumentar la capacidad de búsqueda global sin aumentar la complejidad computacional. Como ya se ha comentado con anterioridad, los algoritmos ACO presentan a veces la desventaja de no obtener un óptimo local debido a que la búsqueda cesa de forma prematura a medida que la cantidad de feromona va en aumento. Por otro lado, la población de hormigas se mantiene constante a lo largo de todas iteraciones, lo que supone un gasto de computación innecesaria en las fases finales del algoritmo ya que la población se mantiene muy grande y el espacio de búsqueda va decreciendo. En el algoritmo PDACO la población de la colonia es mucho más grande en las primeras iteraciones para aumentar así el rango de búsqueda. Posteriormente, a medida que se realizan iteraciones, la población va descendiendo para así disminuir la complejidad computacional. Esta propuesta se combina con diferentes algoritmos ACO, como MMAS (PDMMAS) y ACS (PDACS), para mejorar su funcionamiento. No modifica el algoritmo original ya que únicamente controla la población de la colonia.

4. Resultados

Uno de los resultados que se obtuvo durante la realización de experimentos con el algoritmo AS original fue la presencia de una sinergia en la interacción entre las hormigas. De hecho la calidad de la solución obtenida aumentaba al aumentar también el número de hormigas trabajando en el problema, hasta alcanzar un punto óptimo. Un segundo resultado obtenido fue la comprobación de la viabilidad del proceso de autocatálisis que desempeñaban las hormigas como metodología para la optimización y el aprendizaje. Esto es debido a que el proceso autocatalítico de una hormiga suele converger rápidamente en una mala solución subóptima pero, por otro lado, la interacción de muchos procesos autocatalíticos puede producir la rápida convergencia en un subespacio de soluciones, dentro del cual se buscará posteriormente la mejor solución.

Por otro lado y tras varios estudios y pruebas computacionales, se llegó a la conclusión de que, a pesar de ofrecer resultados satisfactorios en la resolución del problema VRP, el algoritmo AS se comportaba peor que otras heurísticas más estudiadas y mejoradas hasta la fecha, como por ejemplo las listas tabú.

En cuanto al algoritmo ACS, diversos estudios han demostrado que resulta muy eficiente a la hora de resolver problemas de la clase VRP, desde el caso estático (VRP con ventanas de tiempo y/o restricciones de recogida y entrega) hasta el caso dinámico (*on-line* VRP), estudiado por Montemanni et al. (2005). Típicamente, la primera cuestión que se plantea al emplear algoritmos metaheurísticos está relacionada con la convergencia en cuanto a sus posibilidades de encontrar siempre una solución óptima. En relación a los algoritmos ACO más antiguos, y para los dos más eficientes (ACS y MMAS), su convergencia ha sido comprobada por Dorigo y Stützle (2002, 2004). Desafortunadamente, estos resultados no pueden predecir el tiempo que se tardará en encontrar dicha solución óptima. Únicamente se ha presentado un marco analítico, Gutjahr (2006), que permite realizar predicciones teóricas de la velocidad de convergencia de algunos algoritmos ACO específicos.

A modo de comparación cuantitativa, se ha extraído de la literatura una tabla comparativa (Tabla 1) que muestra el rendimiento de los tres algoritmos descritos anteriormente, orígenes para algunas instancias TSP simétricas y asimétricas extraídas de la librería TSPLIB. La comparación está hecha en base al mismo número de rutas construídas para cada algoritmo y los valores indicados se corresponden con la calidad media de la solución para cada

instancia. “opt” indica la solución óptima conocida para cada instancia.

Tabla 1. Comparativa de rendimiento de algoritmos ACO para diversas instancias.

Instancia simétrica	**opt**	**AS**	**ACS**	**MMAS**
ei151	426	437.3	428.1	427.6
kroA100	21282	22471.4	21420.0	21320.3
d198	15780	16702.1	16054.0	15972.5
Instancia asimétrica	**opt**	**AS**	**ACS**	**MMAS**
ft70	38673	39596.3	39099.0	39040.2
kro124p	36230	38733.1	36857.0	36773.5
ftv170	2755	3154.5	2826.5	2828.8

Los resultados computacionales de la Tabla 1 muestran que el algoritmo MMAS consigue el mejor rendimiento. Teniendo en cuenta los resultados obtenidos para el algoritmo AS se puede decir que su desempeño es muy pobre en comparación con el resto. Por otro lado, también es apreciable que la ventaja de MMAS sobre ACS con respecto a la calidad de la solución es más notable en las instancias simétricas que en las asimétricas.

Algunos de los algoritmos ACO propuestos recientemente solventan los inconvenientes que presentan los métodos tradicionales y proporcionan mejores soluciones. Es el caso del M-MMAS y su predecesor MMAS, los cuales se aplican a la instancia Eil51 de la librería TSPLIB para poder realizar una comparativa. Ambos ejecutan las mismas iteraciones y utilizan el mismo número de hormigas. En el caso del M-MMAS se va aumentando el número de elementos (ciudades) mutadas para así también poder comprobar como el número de elementos mutados afecta a la longitud de la ruta de la solución, tal y como se puede comprobar en la Figura 1. La mejor solución conocida para esta instancia es 425. La Figura 1 muestra que la longitud media de la ruta solución de la instancia Eil51 solucionada con M-MMAS es más corta para cualquier número de ciudades mutadas que en el caso de MMAS.

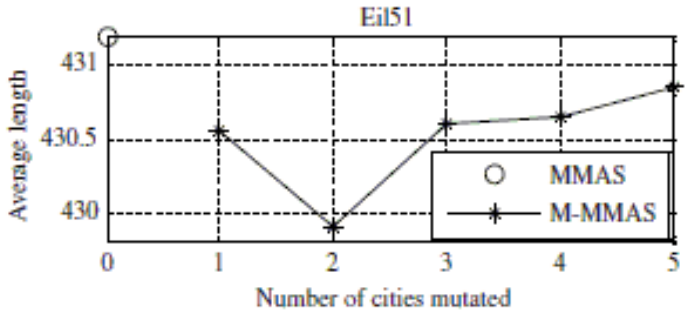

Figura 1. Longitud media de la ruta vs. número de ciudades mutadas en M-MMAS para Eil51.

La segunda instancia utilizada es la St70, cuya mejor solución conocida es 675. Tanto MMAS como M-MMAS ejecutan el mismo número de iteraciones utilizando el mismo número de hormigas. En el caso de M-MMAS se va elevando el número de ciudades mutadas desde 1 hasta 7. La relación entre la longitud media de la ruta y el número de

ciudades mutadas para el caso M-MMAS se muestra en la Figura 2. Se puede apreciar cómo la longitud media de la ruta es menor para todos los casos resueltos con M-MMAS.

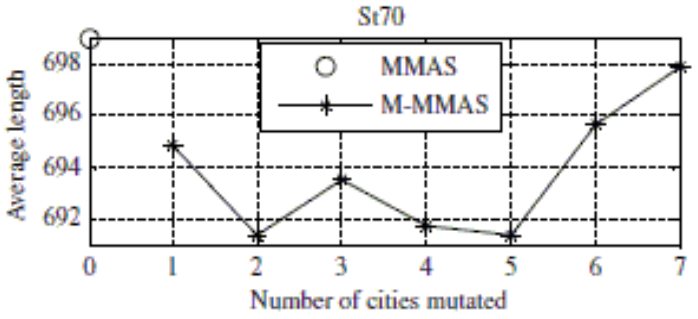

Figura 2. Longitud media de la ruta vs. número de ciudades mutadas en M-MMAS para St70.

Mediante la resolución de estas dos instancias típicas se puede llegar a la conclusión de que el rendimiento del algoritmo M-MMAS para la resolución de problemas de optimización combinatoria es superior al del MMAS. Para la comparación de los algoritmos ACS y M-ACS se realiza el mismo proceso comparativo. La Figura 3 y la Figura 4 muestran las curvas que relacionan la longitud media de las rutas con el número de ciudades mutadas para las dos instancias resueltas. La conclusión obtenida es la misma que para el caso del MMAS y el M-MMAS: el funcionamiento del algoritmo ACS mutado es mejor que el del ACS original. A partir de las figuras también se puede observar que el número de ciudades mutadas no debe ser muy alto ya que sino no se podrá obtener la mejor solución de la mejor solución local debido a que sufrirá un gran cambio. El número adecuado de elementos a mutar deber ser decidido en función de la escala del problema a resolver.

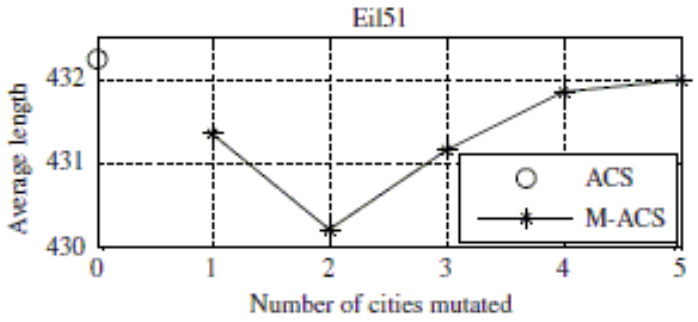

Figura 3. Longitud media de la ruta vs. número de ciudades mutadas en M-ACS para Eil51.

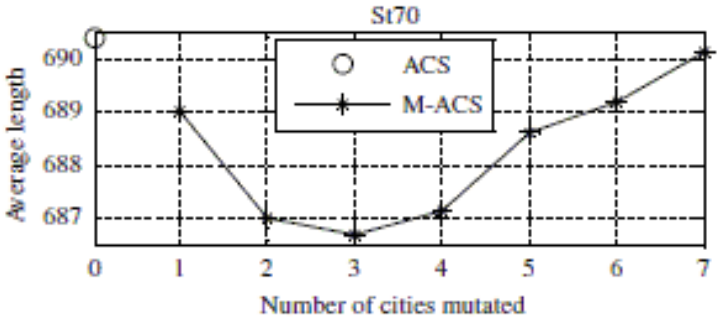

Figura 4. Longitud media de la ruta vs. número de ciudades mutadas en M-ACS para St70.

Otra propuesta mencionada de algoritmo ACO en el que se introducen los operadores genéticos de cruce y mutación es EA. Para demostrar su mejora de rendimiento frente al ya analizado algoritmo ACS, se muestra a continuación algunas simulaciones por ordenador extraídas de la literatura y realizadas sobre instancias de la librería pública TSPLIB. La Figura 6, la Figura 7 y la Figura 8 muestran la comparación de la longitud de ruta de los algoritmos EA y ACS para la resolución de diferentes instancias. La Figura 9, la Figura 10 y la Figura 11 muestran la comparación de la longitud de ruta de los algoritmos EA+NN y ACS+NN. La Figura 12, la Figura 13 y la Figura 5 muestran la comparación de la longitud de ruta para los algoritmos EA+NN, ACS+NN, EA y ACS. La Tabla 2 resume las comparaciones anteriores

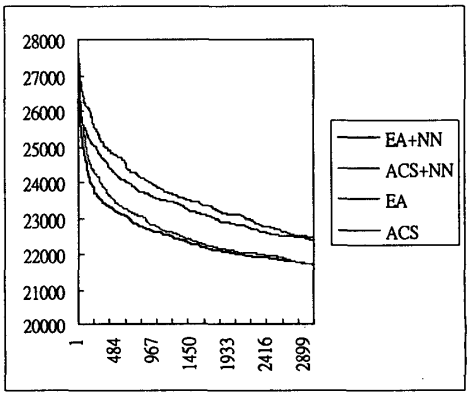

Figura 5. Comparación de longitud de ruta para EA+NN, ACS+NN, EA y ACS para la instancia kroa100.

Tabla 2. Tabla comparativa de las longitudes de ruta de diversos algoritmos.

Problema	ACS	ACS+NN	EA	EA+NN
Eil51	438.3623	432.3149	429.5682	429.5036
Eil76	590.6458	584.7768	573.6383	573.189
Kroa100	22468.93	22388.54	21705.3	21698.45

A partir de los resultados anteriores se puede observar cómo la utilización del algoritmo EA proporciona una mejora significante para la obtención de una solución óptima global o cercana a ella. Es más, según la información mostrada, la adición del método NN para la búsqueda de la solución inicial también mejora el funcionamiento. Finalmene, se puede apreciar que la utilización de EA+NN permite la obtención de mejores soluciones que con el resto de enfoques mencionados.

Otra propuesta de algoritmo comentada anteriormente en este artículo es HK-ACO. Los resultados computacionales mostrados a continuación han sido extraídos de la literatura. La Figura 14 muestra una comparativa entre HK-ACO y una variante del algoritmo AS, Bullnheimer, Hartl y Strauss (1997), para la instancia eil51 extraída de la librería TSPLIB. La óptima longitud de ruta conocida para esta instancia de problema es 426. La imagen muestra como HK-ACO aumenta la convergencia, superando así al algoritmo AS. Por otro lado, la Tabla 3 muestra unos resultados comparativos para instancias de problemas más grandes, extraídos también de otros artículos, entre HK-ACO y el algoritmo MMAS. Para cada uno de ellos se muestran las longitudes de ruta más corta encontradas (mejor) así como las longitudes medias (promedio) para diez iteraciones.

.

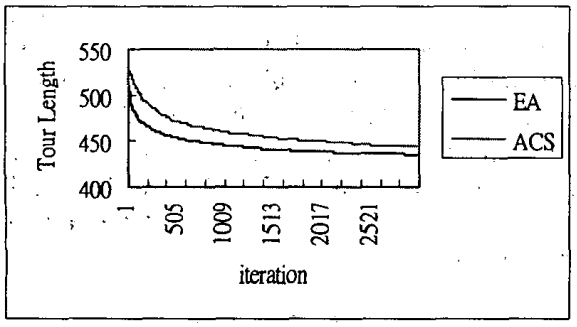

Figura 6. Comparación de longitud de ruta para EA y ACS utilizando la instancia eil51.

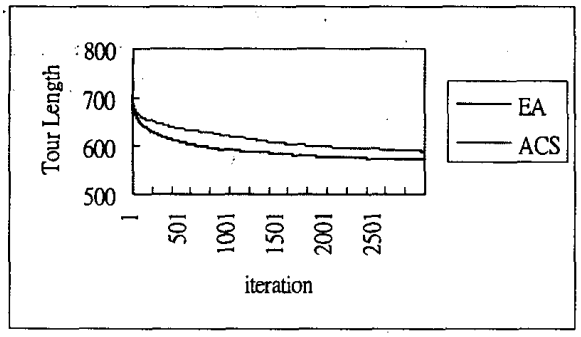

Figura 7. Comparación de longitud de ruta para EA y ACS utilizando la instancia eil76.

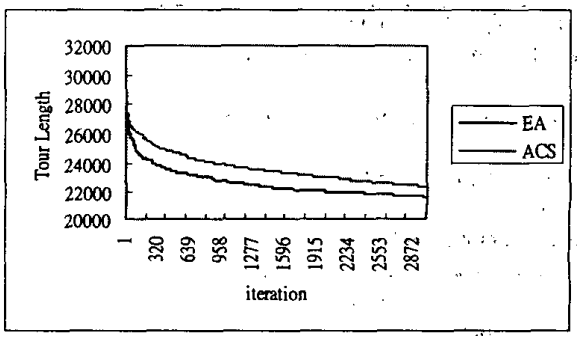

Figura 8. Comparación de longitud de ruta para EA y ACS utilizando la instancia kroa100.

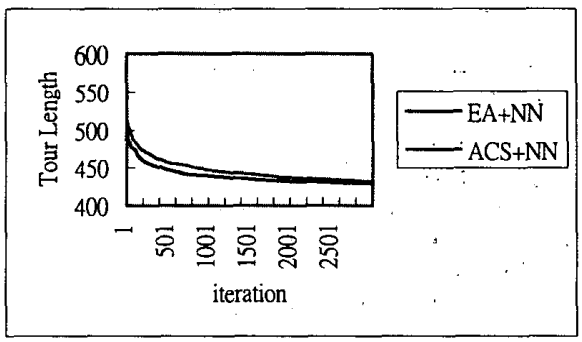

Figura 9. Comparación de longitud de ruta para EA+NN y ACS+NN utilizando la instancia eil51.

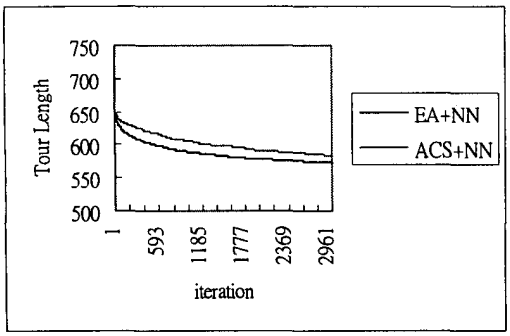

Figura 10. Comparación de longitud de ruta para EA+NN y ACS+NN utilizando la instancia eil76.

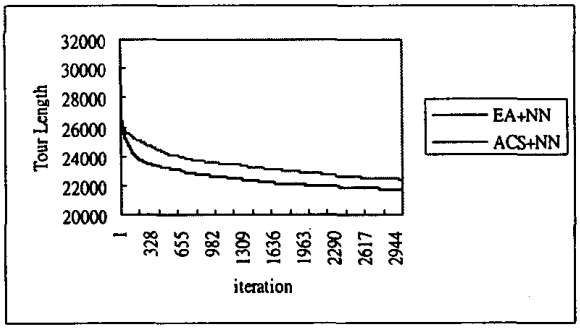

Figura 11. Comparación de longitud de ruta para EA+NN y ACS+NN utilizando la instancia kroa100.

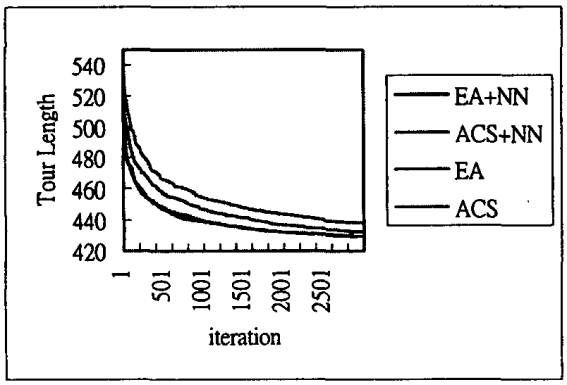

Figura 12. Comparación de longitud de ruta para EA+NN, ACS+NN, EA y ACS para la instancia eil51.

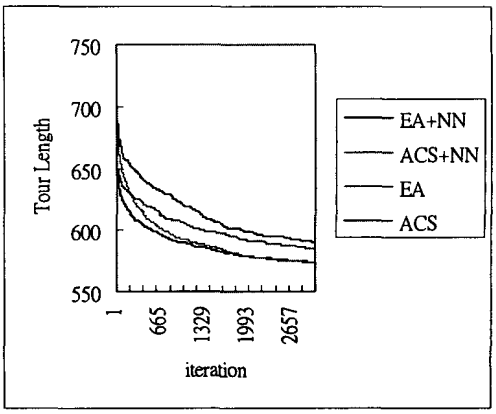

Figura 13. Comparación de longitud de ruta para EA+NN, ACS+NN, EA y ACS para la instancia eil76.

Tabla 3. Soluciones obtenidas utilizando los algoritmos HK-ACO y MMAS para diversas instancias.

Instancia (óptimo)	HK-ACO (mejor)	HK-ACO (promedio)	MMAS (mejor)	MMAS (promedio)
d198 (15780)	15780	15780	15780	15780.9
lin318 (42029)	42029	42051.3	42029	42051.8
att532 (27686)	27686	27702.6	27693	27707.4
pr1002 (259045)	259176	259266.6	259290	260025

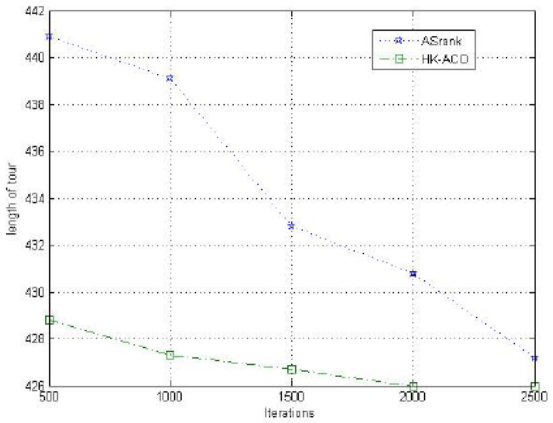

Figura 14. Comparación entre HK-ACO y ASrank para la instancia eil51.

La tabla anterior pone de manifiesto como para algunas instancias (d198 y lin318), la capacidad para encontrar la mejor ruta es la misma para ambos algoritmos, sin embargo, difieren en la estabilidad a la hora de obtenerla. Desde este punto de vista, HK-ACO muestra un carácter estable a la hora de resolver el problema. Para el resto de instancias, la solución generada por HK-ACO es mejor que la generada por MMAS, tanto en precisión como en estabilidad. En conclusión, HK-ACO proporciona mejores resultados que MMAS, hecho que pone de manifiesto que el enfoque que utiliza este algoritmo para ajustar la influencia entre la información heurística y las feromonas es una forma efectiva de mejorar el rendimiento de los algoritmos ACO tradicionales.

Para observar el rendimiento del algoritmo PDACO se han extraído de la literatura algunas gráficas comparativas (Figura 15 y Figura 16), las cuales muestran los resultados obtenidos al resolver una instancia de problema típicamente conocido, utilizando tanto el algoritmo original como el mismo combinado con la técnica de disminución de la población. Se muestra la evolución de la longitud media de ruta obtenida para diferentes tamaños de población inicial en el caso PDACO.

Los resultados de la simulación muestran que al resolver el problema con el algoritmo PDACO, la longitud media de ruta es menor que la obtenida con el algoritmo ACO para todos los casos de población inicial. Por tanto, el rendimiento de algoritmos ACO mejorados

con PDACO es superior al del correspondiente algoritmo ACO original en la resolución de problemas de optimización combinatoria.

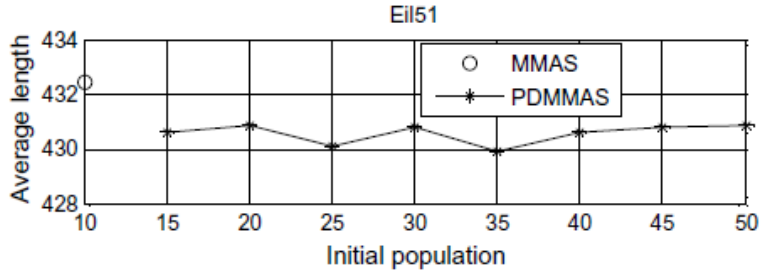

Figura 15. Gráfica comparativa del rendimiento de MMAS y PDMMAS para Eil51.

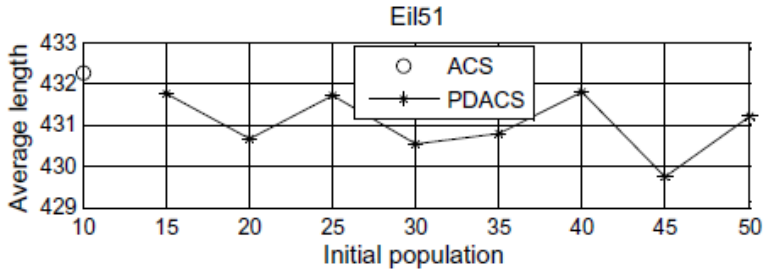

Figura 16. Gráfica comparativa del rendimiento de ACS y PDACS para Eil51.

5. Conclusiones

S*warm intelligence* o inteligencia de enjambre, un relativamente nuevo enfoque para la resolución de problemas, ha sido inspirado por el comportamiento social de algunos insectos y otros animales. Los algoritmos de inteligencia de enjambre están formados por simples individuos que cooperan a través de la autoorganización, es decir, sin ninguna forma de control central sobre los miembros del enjambre.

En particular, las hormigas han inspirado un gran número de métodos y técnicas, entre las cuales, las más estudiadas y exitosas son las técnicas de optimización de propósito general conocidas como optimización con colonias de hormigas (ACO, *Ant Colony Optimization*). Los algoritmos ACO están inspirados en el comportamiento alimentario de algunas especies de hormigas. Estas hormigas depositan feromonas en el suelo para marcar un camino favorable que debe ser seguido por otros miembros de la colonia. Los algoritmos de optimización basados en colonias de hormigas utilizan un mecanismo similar para solventar problemas de optimización, la mayoría de los cuales se tratan de problemas NP-*hard*. Muchos de estos problemas pueden ser clasificados en una de las siguientes categorías: problemas de enrutamiento, como por ejemplo los presentes en la distribución de bienes; problemas de asignación, en los cuales un conjunto de ítems (objetos, actividades, etc.) deben ser asignados a un determinado número de recursos (emplazamientos, agentes, etc.) siguiendo algunas restricciones; problemas de programación temporal, los cuales están relacionados con la asignación de recursos escasos a tareas a lo largo del tiempo; y

problemas de subconjuntos, donde la solución al problema se obtiene mediante una selección de un subconjunto de ítems disponibles. Un punto común a muchas de estas aplicaciones es que los algoritmos ACO con mejor rendimiento utilizan de forma intensiva la fase opcional de búsqueda local.

A día de hoy se han propuesto multitud de variantes de los algoritmos ACO originales, resultando algunas de ellas más exitosas que otras. La optimización basada en colonias de hormigas se trata de una metodología relativamente joven en comparación con otras tales como computación evolutiva, búsquedas tabú o *simulated annealing.* Aún así, se ha demostrado que estos algoritmos son bastante flexibles y eficientes.

La revisión del estado del arte aquí expuesta, centrada en el estudio de la aplicación de algoritmos ACO a problemas TSP y VRP, permite realizar una primera aproximación al planteamiento del problema concreto a resolver que presenta la empresa logística. Algunos de los algoritmos estudiados podrían servir como herramienta para su resolución aunque, debido a la complejidad del problema, se prevé que la solución final estará implementada mediante un algoritmo híbrido que permita alcanzar todos los objetivos previamente descritos.

Referencias

Beckers, R.; Deneubourg, J.L.; Goss, S. (1992). Trails and U-turns in the selection of the shortest path by the ant Lasius Niger. J. Theoretical Biology, Vol. 159, pp. 397–415.

Bullnheimer, B.; Hartl, R.F.; Strauss, C. (1997). A New Rank Based Version of the Ant System: A Computational Study. Central European Journal for Operations Research and Economics, Vol. 7, No. 1, pp. 25-38.

Bullnheimer, B.; Hartl, R.F.; Strauss, C. (1997). Applying the Ant System to the Vehicle Routing Problem. Paper presented at 2nd International Conference on Metaheuristics, Sophia-Antipolis, France.

Bulinheimer, B.; Hartl, R.F.; Strauss, C. (1999). An improved ant algorithm for the vehicle routing problem, Vol. 89, No. 0, pp. 319-328.

Cheng-Fa, T.; Chun-Wei, T. (2002). A New Approach for Solving Large Traveling Salesman Problem Using Evolutionary Ant Rules. Neural Networks IJCNN '02, pp. 1540-1545.

Clarke, G.; Wright, J.W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. Operations Research, Vol. 12, No. 4, pp. 568-581.

Colorni, A.; Dorigo, M.; Maniezzo, V. (1991). Distributed optimization by ant colonies. Proceedings of ECAL-91 – European Conference on Artificial Life, Paris, France, F. Varela and P. Bourgine (Eds.) (pp. 134-142).

Colorni, A.; Dorigo, M.; Maniezzo, V. (1992). An investigation of some properties of an ant algorithm. Proceedins of the Parallel Problem Solving from Nature Conference (PPSN92), Brussels, Belgium, R. Männer, B. Manderick (Eds.) (pp. 509-520).

Colorni, A.; Dorigo, M.; Maniezzo, V. (1991). An autocatalytic Optimizing Process. Technical Report n. 91-016, Politecnico di Milano, Milano, Italy.

Dantzig, G.B.; Ramser, J.H. (1959). The truck dispatching problem. Management Science, Vol. 6, No. 1, pp. 80–91.

Dorigo, M.; Gambardella, L.M. (1997). Ant colony system: A cooperative learning approach to the travelling salesman problem. IEEE Transactions On Evolutionary Computation, Vol.1,

No. 1, pp.53–66.

Dorigo, M.; Stützle, T. (2004). Ant Colony Optimization, MIT Press, Cambridge, MA.

Gutjahr, W.J. (2006). On the finite-time dynamics of ant colony optimization. Methodology and Computing in Applied Probability, Vol. 8, No. 1, pp. 105–133.

Goss, S.; Aron, S.; Deneubourg, J.L.; Pasteels, J.M. (1989). Self-organized shortcuts in the argentine ant. Naturwissenschaften, Vol. 76, pp. 579–581.

Holland, J. H. (1992). Adaptation in natural and artificial systems. Cambridge, MA: MIT.

Kaveh, A.; Shahrouzi, M. (2008). Optimal structural design family by genetic search and ant colony approach. Engineering Computations, Vol. 25, No.3, pp. 268-288.

Lee, Z. L.; Su, S. F.; Chuang, C. C.; Liu, K. H. (2008). Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. Applied Soft Computing Journal, Vol. 8, No. 1, pp. 55-78.

Li, L.; Ju, S.; Zhang, Y. (2008). Improved Ant Colony Optimization for the Traveling Salesman Problem. International Conference on Intelligent Computation Technology and Automation, pp. 76-80.

Montemanni, R.; Gambardella, L.M.; Rizzoli, A.E.; Donati, A.V. (2005). Ant colony system for a dynamic vehicle routing problem. Journal of Combinatorial Optimization, Vol. 10, No. 4, pp. 327–343.

Socha, K.; Knowles, J.; Sampels, M. (2002). A MAX –MIN ant system for the university timetabling problem. Lecture Notes in Computer Science, Vol. 2463, pp. 63-77.

Stützle, T.; Dorigo, M. (2002). A short convergence proof for a class of ACO algorithms. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 4, pp. 358–365.

Stützle, T.; Hoos, H. (2000). MAX-MIN Ant System. Future Generation Computer Systems, Vol. 16, No. 8, pp. 889–914, 2000.

Wu, Z.; Zhao, N.; Ren, G.; Quan, T. (2008). Population declining ant colony optimization algorithm and its applications. Expert Systems with Applications, Vol. 36, No. 2, pp. 6276-6281.

Zhao, N.; Wu, Z.; Zhao, Y.; Quan, T. (2010). Ant colony optimization algorithm with mutation mechanism and its applications. Expert Systems with Applications, Vol. 37, No. 7, pp. 4805-4810.