

Metaheurística de Optimización mediante Colonias de Hormigas y Aplicaciones

Evelyn Menéndez Alonso
evelynma@uclv.edu.cu

Resumen: La mayoría de los Problemas de Optimización Combinatoria de interés científico o práctico están incluidos en la clase NP-completos, ya que no existen algoritmos exactos con complejidad polinómica que permitan resolverlos. Debido a su intratabilidad, se han diseñado una gran cantidad de métodos aproximados, los cuales encuentran buenas soluciones en tiempos razonables. Uno de estos métodos es la metaheurística de Optimización mediante Colonias de Hormigas (ACO); que tiene su fuente de inspiración en el comportamiento de las hormigas reales, que minimizan el recorrido entre su colonia y cualquier fuente de abastecimiento, basándose fundamentalmente en los rastros de feromona que van dejando a su paso. Para la metaheurística ACO se han propuesto varios algoritmos, que desde su surgimiento han probado su amplia aplicabilidad y eficiencia en la solución de Problemas de Optimización Combinatoria.

Palabras Claves: Optimización mediante Colonias de Hormigas, Sistema de Hormigas, Sistema Colonia de Hormigas, Sistema de Hormigas Max-Min, Sistema de Hormigas con Ordenación, Sistema Mejor-Peor Hormiga, ACO en Dos Etapas.

I. INTRODUCCIÓN

La existencia de una gran cantidad y variedad de Problemas de Optimización Combinatoria incluidos en la clase NP-completos que necesitan ser resueltos de forma eficiente, impulsó el desarrollo de procedimientos para encontrar buenas soluciones, aunque no fueran óptimas. Estos métodos, en los que la rapidez del proceso es tan importante como la calidad de la solución obtenida, se denominan heurísticos o aproximados. Un método heurístico es

un procedimiento para resolver un problema de optimización bien definido mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución [1]. Luego, con el propósito de obtener mejores resultados que los alcanzados por los heurísticos tradicionales surgen los denominados procedimientos metaheurísticos. Los procedimientos metaheurísticos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los heurísticos clásicos no son efectivos. Los metaheurísticos proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de la inteligencia artificial, la evolución biológica y los mecanismos estadísticos [2, 3].

Se han desarrollado varias metaheurísticas para solucionar problemas de optimización, entre ellas se encuentran: Búsqueda Tabú [4], Recocido Simulado [5], GRASP [6], Algoritmos Genéticos [7], Optimización mediante Mallas Dinámicas o Dynamic Mesh Optimization (MDO) [8], Optimización basada en Enjambre de Partículas o Particle Swarm Optimization (PSO) [9, 10] y Optimización basada en Colonias de Hormigas [11, 12], de esta última incluida en la categoría de los algoritmos bioinspirados o de vida artificial e inteligencia colectiva [13], trataremos el presente trabajo.

II. METAHEURÍSTICA DE OPTIMIZACIÓN MEDIANTE COLONIAS DE HORMIGAS

La metaheurística Optimización mediante Colonias de Hormigas o Ant Colony Optimization [14], propuesta para resolver problemas complejos de optimización combinatoria, tiene su fuente de inspiración en el comportamiento de colonias de hormigas reales. Las hormigas son capaces de seguir la ruta más corta en su camino de ida y vuelta entre la colonia y una fuente de abastecimiento. Al desplazarse cada una va dejando un rastro de una sustancia química llamada feromona a lo largo del camino seguido, "transmitiéndose información" entre ellas de esta forma [15]. Las feromonas forman un sistema indirecto de comunicación química entre animales de una misma especie, que transmiten información acerca del estado fisiológico, reproductivo y social, así como la edad, el sexo y el parentesco del animal emisor, las cuales son recibidas en el sistema olfativo del animal receptor, quien interpreta esas señales, jugando un papel importante en la organización y la supervivencia de muchas especies [16].

Al iniciar la búsqueda de alimento, una hormiga aislada se mueve a ciegas, es decir, sin ninguna señal que pueda guiarla, pero las que le siguen deciden con buena probabilidad seguir el camino con mayor cantidad de feromona. Considere la Figura 1 donde se observa cómo las hormigas establecen el camino más corto. En la figura (a) las hormigas llegan a un punto donde tienen que decidir por uno de los caminos que se les presenta, lo que resuelven de manera aleatoria. En consecuencia, la mitad de las hormigas se dirigirán hacia un extremo y la otra mitad hacia el otro extremo, como ilustra la figura (b). Ya que las hormigas se mueven aproximadamente a una velocidad constante, las que eligieron el camino más

corto alcanzarán el otro extremo más rápido que las que tomaron el camino más largo, quedando depositada mayor cantidad de feromona por unidad de longitud, como ilustra la figura (c). La mayor densidad de feromonas depositadas en el trayecto más corto hace que éste sea más deseable para las siguientes hormigas y por lo tanto, la mayoría elige transitar por él. Considerando que la evaporación de la sustancia química hace que los caminos menos transitados sean cada vez menos deseables y la realimentación positiva en el camino con más feromona, resulta claro que al cabo de un tiempo casi todas las hormigas transitan por el camino más corto, como se ilustra en la figura (d) [16].

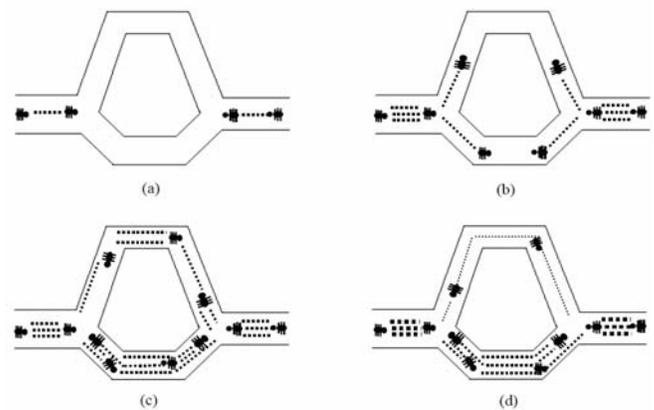


Figura 1: Comportamiento de las hormigas reales.

En analogía con el ejemplo biológico, ACO se basa en la comunicación indirecta de una colonia de agentes simples, llamados hormigas (artificiales), por medio de la huella de feromona (artificial). La huella de feromona en ACO sirve como información numérica distribuida, que las hormigas usan para la construcción probabilística de soluciones del problema a resolver y la adaptan durante la ejecución del algoritmo para reflejar su experiencia de búsqueda [17]. La estructura de un algoritmo genérico de la metaheurística ACO es la siguiente:

```

1 procedimiento metaheurística_ACO()
2  inicialización_de_parámetros
3  mientras (criterio_de_terminación_no_satisfecho)
4    programación_de_actividades
5    hormigas_y_actividad()
6    evaporación_de_feromona()
7    acciones_del_demonio() {opcional}
8  fin programación_de_actividades
9  fin mientras
10 fin procedimiento

1 procedimiento hormigas_y_actividad()
2  repetir en paralelo desde k=1 hasta número_hormigas
3    nueva_hormiga(k)
4  fin repetir en paralelo
5 fin procedimiento

1 procedimiento nueva_hormiga(id_hormiga)
2  inicializa_hormiga(id_hormiga)

```

Las hormigas de la colonia se mueven, concurrentemente y de manera asíncrona, a través de los estados adyacentes de un problema, que puede representarse en forma de grafo con pesos. Este movimiento se realiza siguiendo una regla de transición basada en la información local disponible en las componentes o nodos. Esta información incluye una heurística y una memorística (rastros de feromona) para guiar la búsqueda. La inicialización_de_parámetros depende del algoritmo específico, generalmente deben tenerse en cuenta parámetros como: el rastro inicial de feromona asociado a cada transición o arco, el número de hormigas en la colonia, los pesos que definen la proporción en la que afectarán la información heurística y memorística en la regla de transición probabilística. En programación_de_actividades se controla la planificación de tres componentes: la generación y puesta en funcionamiento de las

```

3  L = actualiza_memoria_hormiga()
4  mientras (estado_actual ≠ estado_objetivo)
5    P = calcular_probabilidades_de_transición(A,L,W)
6    siguiente_estado = aplicar_política_decisión(P,W)
7    mover_al_siguiente_estado(siguiente_estado)
8    si (actualización_feromona_en_línea_paso_a_paso)
9      depositar_feromona_en_el_arco_vistado()
10   fin si
11  L = actualizar_estado_interno()
12 fin mientras
13 si (actualización_feromona_en_línea_a_posteriori)
14  para cada arco_visitado
15    depositar_feromona_en_el_arco_visitado()
16  fin para
17 fin si
18 liberar_recursos_hormiga(id_Hormiga)
19 fin Procedimiento

```

hormigas artificiales; la evaporación de feromona, que se usa como un mecanismo para evitar el estancamiento en la búsqueda y permitir que la hormigas busquen y exploren nuevas regiones del espacio; y las acciones del demonio, utilizadas para implementar tareas desde una perspectiva global que no pueden llevar a cabo las hormigas, por ejemplo, observar la calidad de todas las soluciones generadas y depositar una nueva cantidad de feromona adicional en las transiciones asociadas a algunas soluciones. El procedimiento actualiza_memoria_hormiga() se encarga de especificar el estado inicial desde el que la hormiga comienza su camino y además almacenar la componente correspondiente en la memoria de la hormiga L . La decisión sobre cuál será el nodo inicial depende del algoritmo específico. En los procedimientos calcular_probabilidades_de_transición y aplicar_política_decisión se tienen en consideración el estado actual de la hormiga y el conjunto de arcos del grafo (A), los valores actuales de la feromona visibles

en dicho nodo y las restricciones del problema (W) para establecer el proceso de transición probabilístico hacia otros estados válidos. La actualización_feromona_en_línea_paso_a_paso es el procedimiento donde se actualiza el rastro de feromona asociado a un arco, cuando la hormiga se mueve entre los nodos que este conecta. Una vez que la hormiga ha construido la solución puede reconstruir el camino recorrido y actualizar los rastros de feromona de los arcos visitados mediante el procedimiento llamado actualización_feromona_en_línea_a_posteriori [12, 18].

Se han propuesto varios modelos de la metaheurística ACO, entre ellos se encuentran: Sistema de Hormigas o Ant System (AS) [14], Sistema Colonia de Hormigas o Ant Colony System (ACS) [11], Sistema de Hormigas Max-Min o Max-Min Ant System (MMAS) [19], Sistema de Hormigas con Ordenación o Rank-Based Ant System [20], Sistema Mejor-Peor Hormiga o Best-Worst Ant System [21, 22] y ACO en Dos Etapas o Two-Step Ant Colony Optimization (TS-ACO) [23, 24].

III. ALGORITMOS DE LA METAHEURISTICA ACO

A. Algoritmo Sistema de Hormigas

En el algoritmo Sistema de Hormigas se construyen las soluciones de la siguiente forma: para cada hormiga k en cada paso de construcción se escoge ir del nodo i al siguiente nodo j , $\forall j$, con una probabilidad P_{ij}^k

$$P_{ij}^k = \frac{(\tau_{ij})^\alpha * (\eta_{ij})^\beta}{\sum_{j \in N_i^k} (\tau_{ij})^\alpha * (\eta_{ij})^\beta} \quad \text{si } j \in N_i^k$$

donde: N_i^k es el vecindario alcanzable por la hormiga k cuando se encuentra en el nodo i ; α es el factor de escalado de feromona y β el de visibilidad, ambos se usan para afinar el proceso de búsqueda; τ_{ij} el valor de feromona en el arco que une los nodos i y j ; η_{ij} se denomina función de visibilidad, que depende totalmente de las características del problema que se va a resolver, por ejemplo para el TSP es $1/d_{ij}$, donde d_{ij} es la distancia entre las ciudades i y j . Luego se comparan para todas las hormigas sus soluciones encontradas con la mejor hasta el momento y se modifica esta si alguna de las encontradas la mejora. En la actualización de la huella de feromona se evapora una proporción constante de feromona en cada arco y luego cada hormiga una vez que la solución está completa deposita una cantidad de feromona en dependencia de la calidad de su solución, o sea, actualización en línea a posteriori [14, 25].

B. Algoritmo Sistema Colonia de Hormigas

El algoritmo Sistema Colonia de Hormigas se diferencia del AS en la regla de transición denominada regla proporcional pseudo-aleatoria, que utiliza el parámetro $q_0 \in [0,1]$ y el valor aleatorio $q \in [0,1]$ y calcula P_{ij}^k como sigue:

si $q \leq q_0$

$$P_{ij}^k = \begin{cases} 1, & \text{si } j = \max_{j \in N_i^k} \{ \tau_{ij} * \eta_{ij}^\beta \} \\ 0, & \text{otros casos} \end{cases}$$

si $q > q_0$

$$p_{ij}^k = \frac{(\tau_{ij})^\alpha * (\eta_{ij})^\beta}{\sum_{j \in N_i^k} (\tau_{ij})^\alpha * (\eta_{ij})^\beta} \quad \text{si } j \in N_i^k$$

También existen diferencias en cuanto a la actualización de los rastros de feromona, pues en ACS las hormigas depositan feromona mientras construyen sus soluciones, es decir, actualización en línea paso a paso, que incluye además la evaporación de feromona. Cada vez que una hormiga viaja por un nodo aplica la regla:

$$t_{ij} \leftarrow (1 - \varphi) * t_{ij} + t_0$$

donde $\varphi \in (0,1]$ es un parámetro de decremento de feromona. Por otra parte el demonio actualiza la feromona, o sea, se realiza una actualización de feromona fuera de la línea de los rastros, para esto el algoritmo sólo considera la hormiga que generó la mejor solución global, $S_{mejor-global}$. Esta actualización se hace evaporando primero los rastros de feromona en todas las conexiones utilizadas por la mejor hormiga global y luego depositando feromona en las mismas mediante la regla:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau \quad \forall a_{ij} \in S_{mejor-global}$$

$$\Delta\tau = f(C(S_{mejor-global}))$$

donde a_{ij} es el arco que une los nodos i y j y $C(S_{mejor-global})$ es la calidad de la mejor solución encontrada hasta el momento [11].

C. Algoritmo Sistema de Hormigas MAX-MIN

En el algoritmo Sistema de Hormigas MAX-MIN la probabilidad de ir de un nodo a otro es igual que en el

algoritmo AS. Para explotar la mejor solución durante una iteración o durante la corrida del algoritmo, después de cada iteración, solamente una simple hormiga adiciona feromona, esta puede ser una de las que encuentren la mejor solución de una iteración o la mejor solución global. La regla de modificación de la huella de feromona está dada por:

$$\tau_{ij} \leftarrow \rho\tau_{ij} + \Delta\tau_{ij}^{mejor},$$

donde $\Delta\tau_{ij}^{mejor} = 1/f(s^{mejor})$ y $f(s^{mejor})$ denota el costo de cualquiera de las mejores iteraciones (s^{im}) o la mejor solución global (s^{mg}), ρ es un parámetro entre 0 y 1 que indica la persistencia de la feromona ($1 - \rho$ modela la evaporación). Para evitar el estancamiento de la búsqueda, MMAS impone explícitamente el rango posible del valor de la huella de feromona en cada componente de la solución o arco, esta debe estar limitada para el intervalo $[\tau_{min}, \tau_{max}]$. Después de cada iteración debe asegurarse que se respeten estos límites, si $\tau_{ij} > \tau_{max}$, entonces $\tau_{ij} = \tau_{max}$, si $\tau_{ij} < \tau_{min}$, entonces $\tau_{ij} = \tau_{min}$, además $\tau_{min} > 0$ si $\eta_{ij} < \infty$ para todas las componentes de la solución. Adicionalmente se propone la inicialización de la huella feromona en τ_{max} , consiguiendo una alta exploración de la solución al comienzo del algoritmo [19, 20].

D. Algoritmo Sistema de Hormigas con Ordenación

El algoritmo Sistema de Hormigas con Ordenación [20] es otra extensión del algoritmo AS.

Para este algoritmo las hormigas se ordenan de forma descendente según la calidad de sus soluciones encontradas, por ejemplo: si son m hormigas (S_1, \dots ,

S_m), siendo S_l la mejor solución construida en la iteración actual.

El demonio deposita feromona en las conexiones por las que han pasado las $\sigma - 1$ mejores hormigas, llamadas hormigas elitistas. La cantidad de feromona depositada depende directamente del orden de la hormiga y de la calidad de su solución.

Las conexiones por las que ha pasado la mejor hormiga global reciben una cantidad adicional de feromona que depende únicamente de la calidad de dicha solución. Esta deposición de feromona se considera la más importante, recibe el peso σ .

La regla de actualización de feromona es la siguiente:

$$\tau_{ij} \leftarrow \tau_{ij} + \sigma \Delta \tau_{ij}^{mg} + \Delta \tau_{ij}^{orden}, \text{ donde}$$

$$\Delta \tau_{ij}^{mg} = \begin{cases} f(C(S_{mejor-global})), & \text{si } a_{ij} \in S_{mejor-global} \\ 0, & \text{en otro caso} \end{cases}$$

$$\Delta \tau_{ij}^{orden} = \begin{cases} \sum_{n=1}^{\sigma-1} (\sigma - n)(f(C(S_n))), & \text{si } a_{ij} \in S_n \\ 0, & \text{en otro caso} \end{cases}$$

E. Algoritmo Sistema Mejor-Peor Hormiga

El algoritmo Sistema Mejor-Peor Hormigas [21, 22] incorpora componentes de Computación Evolutiva para mejorar el equilibrio intensificación-diversificación. Mantiene la regla de transición del AS y cambia el mecanismo de actualización de los rastros de feromona, el nuevo mecanismo evapora todos los rastros, refuerza positivamente sólo los de la mejor solución global y negativamente los de la peor solución actual. Aplica una mutación de los rastros de feromona para diversificar y reinicializa la búsqueda cuando se estanca.

La actualización de feromona de la mejor y la peor hormiga se realiza en dos pasos. En el primero se

evaporan todos los rastros de feromona y se aporta en los de la mejor solución global, utilizando la fórmula

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \Delta \tau_{ij}^{mejor-global}.$$

En el segundo paso se realiza una evaporación adicional de los rastros de feromona de la peor solución de la iteración actual que no estén contenidos en la mejor global

$$\tau_{ij} = (1 - \rho)\tau_{ij}, \quad \forall a_{ij} \in S_{peor-actual} \text{ y } a_{ij} \notin S_{mejor-global}.$$

El refuerzo negativo de $S_{peor-actual}$ hace que la regla de actualización tenga un comportamiento más intensificativo.

Este algoritmo considera la búsqueda estancada si durante un número consecutivo de iteraciones (porcentaje del total) no se consigue mejorar la mejor solución global obtenida, en ese caso, se aplica la reinicialización volviendo a poner todos los rastros de feromona a τ_0 .

Para conseguir diversidad en el proceso de búsqueda se mutan los valores de los rastros de feromona, la mutación se aplica en cada rastro de feromona con probabilidad: $\tau_{ij} = \tau_{ij} + N(0, \tau_{umbral})$, donde

$$\tau_{umbral} = \frac{\sum \tau_{ij}}{n}, \quad a_{ij} \in S_{mejor-global},$$

a cada rastro mutado se le añade un valor Normal de media 0 en $[-\tau_{umbral}, \tau_{umbral}]$. τ_{umbral} corresponde a la media de

los rastros de feromona de $S_{mejor-global}$. La función de mutación se caracteriza porque la fuerza de la mutación aumenta con las iteraciones, si τ_{umbral} es

cercano a τ_0 la mutación es pequeña, pero según crecen los rastros de $S_{mejor-global}$, aumenta la mutación.

El algoritmo Sistema Mejor-Peor Hormiga consigue un buen balance entre diversificación e intensificación.

F. Modelo ACO en Dos Etapas

El modelo ACO en Dos Etapas [23, 24], divide el proceso de búsqueda, en la primera etapa se alcanzan soluciones parciales que sirven de estado inicial para la búsqueda en la segunda etapa, de forma que $CM(Ei^*) + CCABH(Ei^*) < CCABH(Ei)$, donde Ei es un estado inicial generado aleatoriamente u obtenido por cualquier otro método sin un costo computacional significativo, Ei^* un estado inicial generado por algún método M que lo acerca al estado final con un costo $CM(Ei^*)$ y $CCABH(x)$ el costo computacional de encontrar una solución desde el estado x usando un algoritmo de búsqueda heurística ABH.

La división entre etapas se logra dando valores distintos a algunos de los parámetros del algoritmo. Para fijar estos valores a los parámetros en cada etapa se establece un factor de proporcionalidad r que indica en que medida cada etapa abarca el proceso de búsqueda completo. Este factor está en el intervalo $(0,1)$, $0 < r < 1$. Por ejemplo, si $r=0.3$, esto significa que el primer paso cubrirá el 30% del proceso de búsqueda y el segundo paso el resto.

En los algoritmos basados en colonias de hormigas los parámetros principales que definen el tamaño de la exploración son: la cantidad de hormigas (m), número de ciclos (nc), tamaño de la solución esperada (nn) y tiempo de ejecución. Los valores de estos parámetros en cada etapa pueden ser calculados según las expresiones:

Primera etapa

$$m_1 = r * m$$

$$nc_1 = r * nc$$

$$nn_1 = r * nn$$

Segunda etapa

$$m_2 = m - m_1$$

$$nc_2 = nc - nc_1$$

$$nn_2 = nn - nn_1$$

En el proceso desarrollado en la primera etapa se almacenan las mejores subsoluciones que servirán de estados iniciales para la búsqueda en la segunda etapa.

IV. APLICACIONES DE ACO

Los algoritmos de ACO se han aplicado con resultados eficientes a diversos problemas de optimización combinatoria, tanto estáticos como dinámicos. Generalmente los algoritmos de ACO se combinan con algoritmos de búsqueda local que refinan las soluciones encontradas.

Entre las principales aplicaciones se encuentran: Problema del Viajero Vendedor o Travelling Salesman Problem (TSP) [11, 14]; Problema de Asignación Cuadrática o Quadratic Assignment Problem (QAP) [26-28]; Cubrimiento de Conjuntos o Set Covering Problem (SCP) [29]; Secuenciación de Tareas o Job Shop Scheduling (JSS) [30]; Enrutamiento de Redes de Comunicaciones [31]; Enrutamiento de vehículos [32]; Ordenación Secuencial [33]; Secuenciación o Flow Shop Scheduling (FSS) [34]; Coloreado de Grafos [35]; Aprendizaje Automático o Machine Learning, principalmente en el diseño de algoritmos de aprendizaje para estructuras de representación del conocimiento: reglas clásicas [36, 37], reglas difusas [38, 39] y reglas bayesianas [40, 41] y Diseño de Circuitos Lógicos Combinatorios [42].

V. CONCLUSIONES

Presentamos los aspectos teóricos relacionados con la metaheurística de Optimización mediante Colonias de Hormigas. Explicamos las variantes algorítmicas:

Sistema de Hormigas, Sistema de Colonia de Hormigas, Sistema de Hormigas Max-Min, Sistema de Hormigas con Ordenación, Sistema Mejor-Peor Hormiga y el modelo ACO en Dos Etapas.

Expusimos las principales aplicaciones de esta metaheurística.

REFERENCIAS

- [1] Díaz, A., et al., eds. *Optimización Heurística y Redes Neuronales*. 1996, Paraninfo S.A: Madrid, España.
- [2] Osman, I.H. and J.P. Kelly, eds. *Meta-Heuristics: Theory and Applications*. 1996, Kluwer Academic: Boston.
- [3] Martí, R., *Algoritmos heurísticos en optimización combinatoria*, in *Departamento de Estadística e Investigación Operativa, Facultad de Ciencias Matemáticas*. 2003, Universidad de Valencia: España.
- [4] Glover, F., *Tabu search*. Part I. *ORSA Journal on Computing*, 1989. 1(3): p. 190-206.
- [5] Kirkpatrick, S., C.D. Gelatt, and M.P. Vecchi, *Optimization by simulated annealing*. *Science*, 1983. 220: p. 671-680.
- [6] Marques-Silva, J.P. and K.A. Sakallah, *GRASP: A Search Algorithm for Propositional Satisfiability*. *IEEE Transactions of Computers*, 1999. 48: p. 506-521.
- [7] Yamada, T. and R. Nakano, *A Genetic Algorithm Applicable to Large-Scale Job Shop Problems*. 2nd International Conference on Parallel Problem Solving from Nature, North-Holland, Amsterdam, 1992: p. 281-290.
- [8] Puris, A. and R. Bello, *Optimización basada en Mallas Dinámicas. Su aplicación en la solución de problemas de optimización continuos*. VI Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB'09), 2009.
- [9] Eberhart, R.C. and J. Kennedy, *A new optimizer using particle swarm theory*. *Proceedings of the Sixth International Symposium on Micromachine and Human Science*. Nagoya, Japan, 1995: p. 39-43.
- [10] Adly, A.A. and S.K. Abd-El-Hafiz, *Field computation in non-linear magnetic media using particle swarm optimization*. *Journal of Magnetism and Magnetic Materials*, 2004. In Press, Uncorrected Proof
- [11] Gambardella, L.M. and M. Dorigo, *Ant Colony System: A cooperative learning approach to the traveling salesman problem*. *IEEE Transactions on Evolutionary Computation* 1997. 1(1): p. 53-66.
- [12] Dorigo, M. and G. Di Caro, *The Ant Colony Optimization meta-heuristic*, in *New Ideas in Optimization*, D. Corne, M. Dorigo, and F. Glover, Editors. 1999, McGraw-Hill: London, UK. p. 11-32.
- [13] Rasmussen, S. and M.J. Raven, *Collective Intelligence of the Artificial Life Community on Its Own Successes, Failures, and Future*. *Artificial Life*, Massachusetts Institute of Technology, 2003. 9: p. 207-235.
- [14] Dorigo, M., V. Maniezzo, and V.M. Coloni, *The Ant System: Optimization by a Colony of Cooperating Agents*, in *IEEE. 1996: Transactions on Systems, Man, and Cybernetics-Part B*. p. 29-41.
- [15] Barcos, L., et al., *Algoritmo basado en la optimización mediante colonias de hormigas para la resolución del problema del transporte de carga desde varios orígenes a varios destinos.*, in *V Congreso de Ingeniería del Transporte*. 2002: Santander - CIT. p. 9.
- [16] Barán, B. and M. Almirón, *Colonias distribuidas de hormigas en un entorno paralelo asíncrono.*, in *XXVI Conferencia Latinoamericana de Informática*. 2000: CLEI'00, México.
- [17] Dorigo, M. and T. Stützle, eds. *The Ant Colony Optimization Metaheuristic: Algorithms, Applications and Advances*, in *Metaheuristics Handbook*. International

- Series in Operations Research and Management Science, ed. F. Glover and G. Kochenberger. 2001, Kluwer Academic Publishers.
- [18] Alonso, S., et al., *La Metaheurística de Optimización Basada en Colonias de Hormigas: Modelos y Nuevos Enfoques*. Proyecto “Mejora de Metaheurísticas mediante Hibridación y sus Aplicaciones” Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S. Ingeniería Informática. Universidad de Granada, España, 2003.
- [19] Stützle, T. and H.H. Hoos, *MAX-MIN Ant System*. Future Generation Computer Systems, 2000. 16(8): p. 889-914.
- [20] Bullnheimer, B., R.F. Hartl, and C. Strauss, *A new rank-based version of the Ant System: A computational study*. Central European Journal for Operations Research and Economics, 1999. 1:7: p. 25-38.
- [21] Cerdón, O., F. Herrera, and L. Moreno., *Integración de Conceptos de Computación Evolutiva en un Nuevo Modelo de Colonia de Hormigas*. Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA'99), 1999: p. 98-104.
- [22] Cerdón, O., et al., *A New ACO Model Integrating Evolutionary Computation Concepts: The Best-Worst Ant System*. From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms, 2000: p. 22-29.
- [23] Bello, R. and A. Puris, *Two Step Ant Colony Systems to Solve the Feature Selection Problem*. 11th Iberoamerican Congress on Pattern Recognition, CIARP. Mexico, 2006: p. 588-596.
- [24] Puris, A. and R. Bello, *Two-Step Ant Colony Optimization for solving the Traveling Salesman Problem*. 2nd International work-conference on the interplay Between natural and artificial computation. España, 2007 p. 307-316.
- [25] White, T., S. Kaegi, and T. Oda, *Revisiting Elitism in Ant Colony Optimization*. Genetic and Evolutionary Computation Conference, 2003.
- [26] Stützle, T. and M. Dorigo, eds. *ACO algorithms for the quadratic assignment problem*. New Ideas in Optimization. 1999, McGraw-Hill.
- [27] Colomi, V.M., *The Ant System applied to the Quadratic Assignment Problem*, in *Transactions on Knowledge and Data Engineering*, in *IEEE* 1999: EEUU.
- [28] Demirel, N.C. and M.D. Toksari, *Optimization of the quadratic assignment problem using an ant colony algorithm*. Applied Mathematics and Computation, 2006. 183 (1): p. 427-435.
- [29] Silva, R.M.A. and G.L. Ramalho, *Ant system for the set covering problem*. Systems, Man, and Cybernetics. IEEE International Conference, 2001. 5: p. 3129-3133.
- [30] van der Zwaan, S. and C. Marques, *Ant Colony Optimisation for Job Shop Scheduling*. Proceedings of the Third Workshop on Genetic Algorithms and Artificial Life (GAAL 99), 1999.
- [31] Schoonderwoerd, R., et al., *Ant-based load balancing in telecommunications networks*. Adaptive Behavior, 1996. 5: 2: p. 169-207.
- [32] Bullnheimer, B., R.F. Hartl, and C. Strauss, *An improved ant system algorithm for the vehicle routing problem*, in *Annals of Operations Research*. 1999. p. 319-328.
- [33] Gambardella, L.M. and M. Dorigo, *Ant Colony System hybridized with a new local search for the sequential ordering problem*. INFORMS Journal on Computing, 2000. 12: 3: p. 237-255.
- [34] Stützle, T., *An ant approach to the flow shop problem*. Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98) . Verlag Mainz, Wissenschaftsverlag, Aachen, Alemania, 1998. 3: p. 1560-1564.
- [35] Costa, D. and A. Hertz, *Ant can colour graphs*. Journal of the Operational Research Society, 1997. 48: p. 295-305.
- [36] Parpinelli, R.S., H.S. Lopes, and A.A. Freitas, *An ant colony based system for data mining: application to medical data*. Proceedings of the Genetic and Evolutionary

- Computation Conference (GECCO-2001). Morgan Kaufmann Publishers, San Francisco, CA, 2001: p. 791-798.
- [37] Parpinelli, R.S., H.S. Lopes, and A.A. Freitas, *Data mining with an Ant Colony Optimization algorithm*. IEEE Transaction on Evolutionary Computation, 2002. 6: 4: p. 321-332.
- [38] Casillas, J., O. Cordón, and F. Herrera, *Learning fuzzy rules using ant colony optimization algorithms*. International Workshops on Ant Algorithms. Université Libre de Bruxelles, Belgium, 2000: p. 13-21.
- [39] Casillas, J., O. Cordón, and F. Herrera, *Different approaches to induce cooperation in fuzzy linguistic models under the COR methodology*, in *Technologies for Constructing Intelligent Systems 1*, Physica-Verlag, Editor. 2002.
- [40] Campos, L.M.d., J.A. Gámez, and J.M. Puerta, *Learning bayesian networks by ant colony optimisation: searching in two different spaces*. Mathware & Soft Computing, 2002. 9: 2-3: p. 251-268.
- [41] Campos, L.M.d., et al., *Ant colony optimization for learning bayesian networks*. International Journal of Approximate Reasoning, 2002. 31: 3: p. 291-311.
- [42] Mendoza, B. and C.A. Coello, *An approach based on the use of the Ant System to design combinatorial logic circuits*. Mathware & Soft Computing, 2002. 9: 2-3: p. 235-250.