Aplicación del Algoritmo de Optimización basado en Colonia de Hormigas (ACO) al Problema del Viajante (TSP)

Alejandro Muñoz Navarro
Universidad Politécnica de Madrid
alejandro.mnavarro@alumnos.upm.es

4 de julio de 2021

Resumen

En este documento se ha recopilado toda la información sobre la aplicación del Algoritmo de Optimización basado en Colonia de Hormigas (ACO) al Problema del Viajante (TSP). En este trabajo se han usado algunos ejemplos de la página web [1] y el lenguaje de programación Python 3 para desarrollar una aplicación que pueda resolver el problema. Además, se han comparado diferentes resultados según los parámetros, para llegar a la solución más cercana a la óptima.

1. Introducción

Propuesto por Marco Dorigo en 1992 en su tesis doctoral [2], el algoritmo de Optimización basado en la Colonia de Hormigas o Ant Colony Optimization (ACO) ha logrado destacar entre los numerosos algoritmos de optimización.

Aunque el algoritmo tenía inicialmente como objetivo buscar un camino óptimo en un grafo, basándose en el comportamiento de las hormigas que buscan un camino entre su colonia y una fuente de alimento, la idea original se ha diversificado desde entonces para resolver múltiples problemas numéricos. Este algoritmo hace uso de hormigas artificiales, las cuales son métodos multiagentes inspirados en el comportamiento de las hormigas reales. La comunicación basada en feromonas de las hormigas biológicas suele ser el paradigma predominantemente utilizado. Las combinaciones de hormigas artificiales y algoritmos de búsqueda local se han convertido en un método de elección para numerosas tareas de optimización que implican algún tipo de grafo, por ejemplo, el enrutamiento de vehículos y el enrutamiento de Internet.

En este trabajo se ha aplicado este algoritmo al Problema del Viajante o Travel Salesman Problem (TSP). Este problema responde a la siguiente pregunta: dada una lista de ciudades y las distancias entre cada una de ellas, ¿cuál es la ruta más corta posible para poder visitar cada ciudad exactamente una vez y al finalizar regresar a la ciudad de origen? Para ello, se han usado ejemplos obtenidos de la página web [1] con el objetivo de acercarnos a la solución más optimizada del problema.

En las siguientes secciones se detallará el problema, el algoritmo utilizado junto con la selección de parámetros, la comparación de resultados y finalmente se expondrá la solución más óptima obtenida.

2. Problema del Viajante

Este problema fue formulado por primera vez en 1930 y es uno de los problemas de optimización más estudiados. Es usado como prueba para muchos métodos de optimización. Como se puede observar en la figura 1, en el problema se presentan diferentes ciudades por las que el viajero deberá pasar al menos una vez. Además, cada arco del grafo o camino entre ciudades, consta de un peso. Por lo que el objetivo es encontrar un camino con un peso mínimo.

El problema presenta N! rutas posibles que el viajero podría tomar, aunque se podría simplificar ya que dada una ruta nos da igual el punto de partida y esto reduce el número de rutas a examinar en un factor N, quedando (N-1)!. Además, como no importaría la dirección en que se desplazase el viajante, el número de rutas a examinar se reduciría a la mitad. Por lo tanto, debemos considerar que el viajero podría tomar (N-1)!/2 rutas posibles.

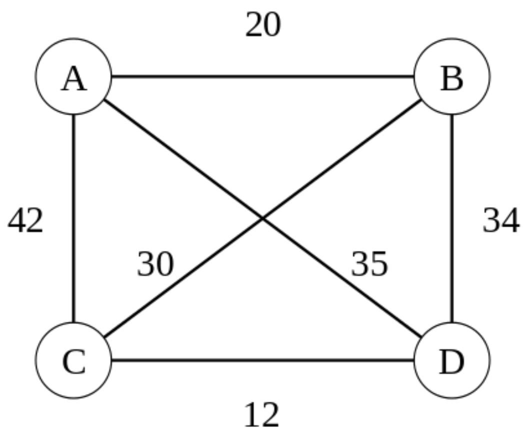

Figura 1: TSP simétrico con 4 ciudades

3. Algoritmo de Optimización basado en la Colonia de Hormigas

Mencionado anteriormente, el algoritmo de optimización basado en la colonia de hormigas ha sido seleccionado para la realización de este trabajo debido a su gran potencial en la búsqueda del camino más óptimo en el problema del viajero (TSP).

Debemos recordar que las hormigas son prácticamente ciegas, y sin embargo, moviéndose prácticamente al azar, acaban encontrando el camino más corto desde su nido hasta la fuente de alimentos más cercana. Pero es importante tener en cuenta algunas consideraciones:

- Una sola hormiga no es capaz de realizar todo el trabajo.
- Una hormiga cuando se mueve deja un rastro de feromonas, para que las demás puedan seguirla.

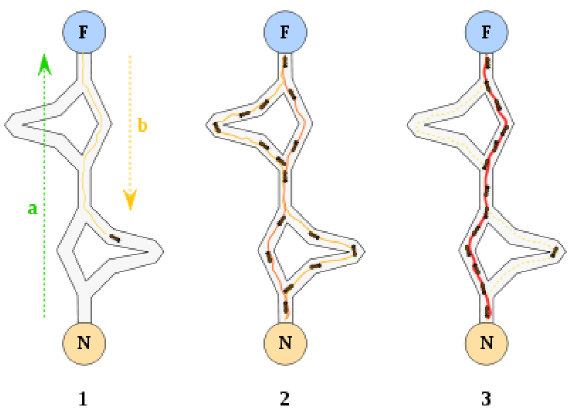

Figura 2: Comportamiento Colectivo de las Hormigas

Los siguientes pasos explican, de una forma intuitiva, porqué la forma de moverse de las hormigas hace lograr alcanzar un camino mínimo entre el nido y la fuente de alimentos:

1. Una hormiga exploradora se mueve de manera aleatoria alrededor de la colonia.
2. Si esta encuentra una fuente de comida, retorna a la colonia de manera más o menos directa, dejando un rastro de feromonas.
3. Estas feromonas atraen a las hormigas más cercanas.
4. Al regresar a la colonia con alimentos estas hormigas depositan más feromonas, por lo que fortalecen las rutas de conexión.
5. Si existen dos rutas para llegar a la misma fuente de alimentos, la ruta más corta será recorrida por más hormigas que la ruta más larga. Por lo que la ruta más corta aumentará en mayor número de feromonas y será más atractiva para las siguientes hormigas.
6. La ruta más larga irá desapareciendo debido a que las feromonas se evaporan.
7. Finalmente, todas las hormigas habrán escogido el camino más corto.

Adaptando este procedimiento al Problema del Viajero, tal y como se muestra en la figura 3, en cada etapa la hormiga eligirá moverse de una ciudad a otra de acuerdo con algunas reglas:

1. Visitar cada ciudad exactamente una vez.
2. Una ciudad a una mayor distancia tiene menos probabilidades de ser elegida.
3. Cuanto más intenso sea el rastro de feromonas trazado entre dos ciudades, mayor será la probabilidad de que se elija ese borde.
4. Si una hormiga completa su viaje y el camino es corto, esta deposita feromonas en todo el recorrido.
5. Después de cada iteración, los rastros de feromonas se evaporan.

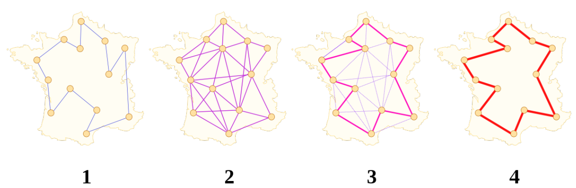

Figura 3: Algoritmo de Optimización de Colonia de Hormigas

3.1. ACO para el Problema del Viajante (TSP)

Para la implementación de este algoritmo con el objetivo de dar una solución optima al problema del viajante, se han definido previamente algunos parámetros fundamentales. A continuación describiremos cada uno de los parámetros de los que consta este algoritmo:

- m: Cantidad de hormigas en cada iteración.
- Iteraciones: Número de repeticiones de la aplicación.
- ρ: Factor de evaporación de las feromonas.
- Q: Cantidad de feromonas liberadas en cada iteración.
- α y β: Son dos parámetros ajustables que controlan el peso relativo de cada una de las medidas en la heurística resultante.

Inicialmente, cada una de las hormigas se repartirán aleatoriamente por las ciudades disponibles siendo estos los puntos iniciales. Siguiendo los artículos [3] y [4], cada una de ellas dejará un rastro de feromonas inicial $\rho_0 = \frac{m*Q}{N*L_0}$ donde L_0 es la longitud del recorrido producido por la heurística del vecino más cercano y N es el número de ciudades. De tal forma que cada hormiga elegirá un camino u otro en función de la cantidad de feromonas y la distancia entre esos dos puntos. Para calcular el posible camino de una hormiga, se utiliza la ecuación 1, donde τ_{ij} son las feromonas depositadas por ese camino y η_{ij} es la visibilidad del camino cuyo valor viene dado por la ecuación 2.

$$p_{ij}^k = \frac{\tau_{ij}^\alpha \cdot \eta_{ij}^\beta}{\sum_{l \in N_i^k} \tau_{ij}^\alpha \cdot \eta_{ij}^\beta} \tag{1}$$

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{2}$$

Además, se han definido dos parámetros α y β con el objetivo de ajustar cada una de estas medidas. En caso de ser $\alpha = 0$, las ciudades más cercanas en cada paso son las que tienen mayor probabilidad de ser seleccionadas. Y en consecuencia, las hormigas no usan el conocimiento de la colonia para mejorar su comportamiento, que viene definido por la cantidad de feromonas que hay en las aristas. Por otro lado, si $\beta = 0$, únicamente interviene la feromona, lo que puede llevar a recorridos no muy buenos y sin posibilidad de mejora.

Una vez la hormiga ha llegado al destino, se debe actualizar las feromonas de cada camino siguiendo la ecuación 3 y 4, respecto al desgaste o evaporación ρ y si la hormiga pasó dejando feromonas. Siendo Q el parámetro de aprendizaje y L_k el coste del camino por cada hormiga.

$$\tau_{ij} = (1-\rho)\tau_{ij} + \sum \Delta\tau_{ij} \tag{3}$$

$$\Delta\tau_{ij} = \begin{cases} Q/L_k & \text{si se uso ij} \\ 0 & \text{en otro caso} \end{cases} \tag{4}$$

4. Resultados

La simulación de este algoritmo se ha realizado definiendo diferentes valores según el artículo [3] para cada uno de los parámetros mencionados anteriormente (Tabla 1). Sin embargo, debido a la potencia necesaria para poder realizar un gran número de pruebas con diferentes parámetros siguiendo ese artículo, es decir 1250 iteraciones, se ha reducido estas a la mitad (625). Además se ha tenido en cuenta que algunos parámetros necesitan estar dentro de un rango ($0 \leq \rho \leq 1$).

m	5	10	20
ρ	0.15	0.25	0.5
Q	0.3	0.6	0.9
α	0.05	0.1	0.35
β	0.5	1	2

Cuadro 1: Valores de losParámetros

El objetivo es realizar pruebas con la combinación de cada uno de ellos para lograr obtener un resultado óptimo. Debido a la gran cantidad de resultados, mostraremos únicamente los seis más óptimos. Como se aprecia en las figuras 4, 5, 6, 7, 8 y 9, los resultados obtenidos no distan mucho unos de otros. La gran mayoría de resultados son con un número de 20 hormigas, lo cual no nos sorprende, ya que con un mayor número de hormigas, el algoritmo tiene mayor probabilidad de converger a valores más óptimos. En cuanto a los demás parámetros, los resultados más óptimos se han obtenido con un $\rho = 0{,}25$, un $\alpha = 0{,}1$ y un $\beta = 1$. En cuanto a Q, los resultados más óptimos se han definido en $Q = 0{,}3$ y en $Q = 0{,}6$. Comparando con el artículo [3], los parámetros que nos han dado los resultados más óptimos con un número menor de iteraciones distan de los parámetros usados en dicho artículo. El número de hormigas se ha incrementado y α se ha conservado, mientras que β y Q se han reducido.

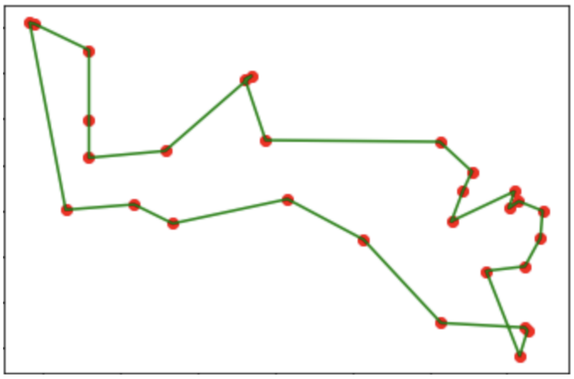

Figura 4: $m = 5, \rho = 0{,}25, \alpha = 0{,}1, Q = 0{,}3, \beta = 1$. Distancia = 29881.981999188767

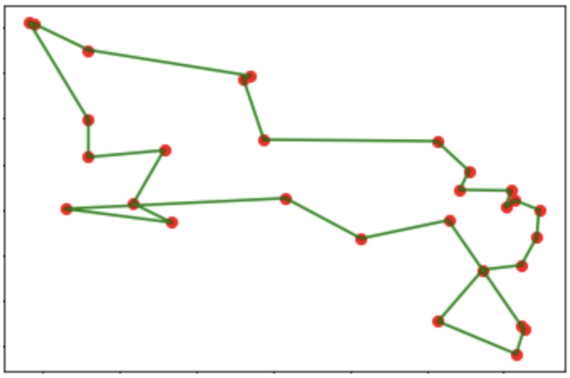

Figura 5: $m = 5, \rho = 0{,}5, \alpha = 0{,}05, Q = 0{,}3, \beta = 1$. Distancia = 29863.25398663174

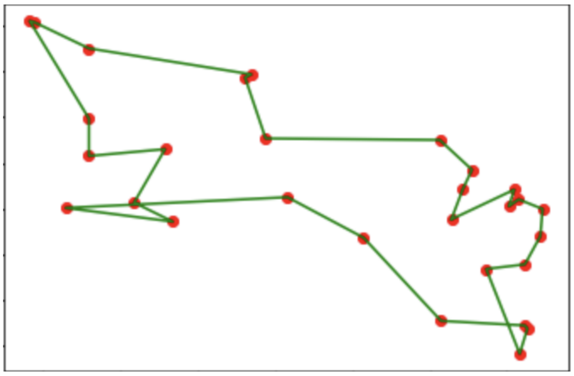

Figura 6: $m = 10, \rho = 0{,}25, \alpha = 0{,}05, Q = 0{,}6, \beta = 2$. Distancia = 29759.858615278194

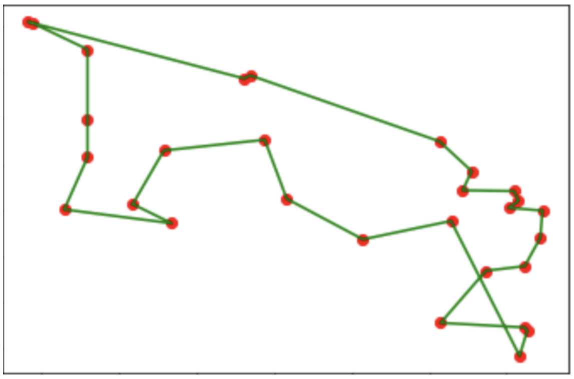

Figura 7: $m = 20, \rho = 0{,}25, \alpha = 0{,}35, Q = 0{,}6, \beta = 1$. Distancia = 29469.080386194644

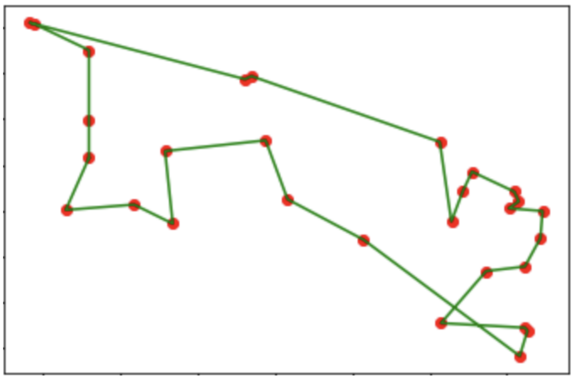

Figura 8: $m = 20, \rho = 0{,}25, \alpha = 0{,}1, Q = 0{,}3, \beta = 2$. Distancia = 29954.985818037243

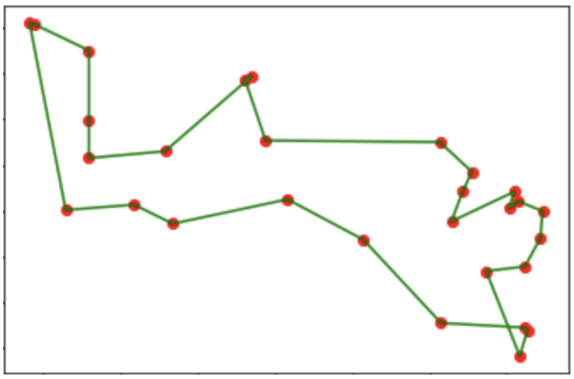

Figura 9: $m = 20, \rho = 0{,}5, \alpha = 0{,}1, Q = 0{,}6, \beta = 1$. Distancia = 29901.441524353922

5. Conclusiones

Comparando con la web [1] de la cual se ha obtenido el mapa, el resultado para *"WI29 - Western Sahara"* estaba definido en una distancia óptima de 27603 [10]. Sin embargo, nuestro algoritmo ha logrado una distancia óptima de 29469.080386194644, en el caso de la figura 7. Esta distancia se podría haber mejorado probablemente con un número mayor de iteraciones, pero para la comparación de nuestro problema se requería de una gran capacidad de cómputo al tratarse de un número muy alto de ejecuciones. No obstante, al tratarse de un algoritmo que parte de un punto inicial aleatorio, sería muy difícil de comprobar si este habría mejorado con un mayor número de iteraciones, puesto que este se comportará distintamente.

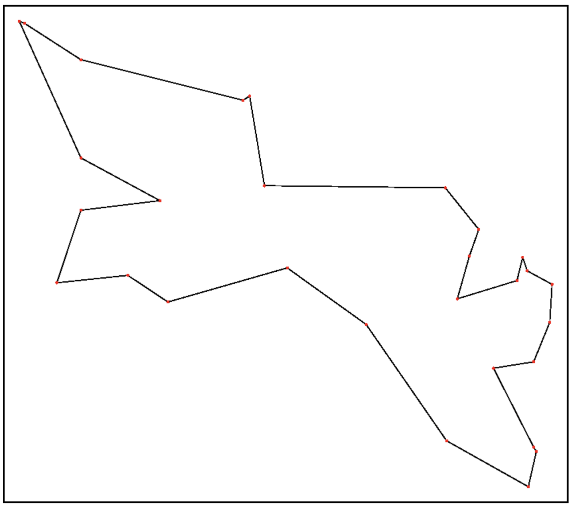

Figura 10: Camino óptimo *"WI29 - Western Sahara"* [1]. Distancia = 27603

Referencias

[1] Tsp test data. http://www.math.uwaterloo.ca/tsp/data/.

[2] M. Dorigo. *Optimization, Learning and Natural Algorithms*. Politecnico di Milano, Italy, 1992.

[3] M. Dorigo and L. M. Gambardella. Ant colonies for the travelling salesman problem. *Biosystems*, 43(2):73–81, 1997.

[4] M. Dorigo and T. Stützle. *Ant Colony Optimization Algorithms for the Traveling Salesman Problem*, pages 65–119. 2004.