Problema del Viajante Resuelto con el Bioalgoritmo de las Luciérnagas

Brainiac

Octubre 22 del 2024

Introducción:

El Problema del Viajante (TSP) es uno de los problemas más estudiados en la teoría de la optimización combinatoria. En este problema, el objetivo es encontrar la ruta más corta posible que permita a un viajante visitar un conjunto de ciudades exactamente una vez y regresar al punto de origen. Este problema es NP-difícil, lo que significa que no existe un algoritmo eficiente conocido para resolverlo en tiempo polinómico para todas las instancias.

El TSP tiene aplicaciones prácticas en logística, planificación de rutas y diseño de circuitos, entre otros campos. Su dificultad proviene del gran número de posibles rutas, que crece exponencialmente con el número de ciudades. Por lo tanto, encontrar una solución óptima es un desafío significativo.

¿Por qué este problema y no otro?

El TSP ha sido elegido por varias razones:

- Relevancia Práctica: La optimización de rutas es crucial en áreas como la logística y la planificación de rutas de transporte. Resolver el TSP puede mejorar la eficiencia en la gestión de recursos y reducir costos operativos.

- Desafío Computacional: El TSP es un problema bien conocido en la teoría de la complejidad computacional. Su naturaleza NP-difícil lo convierte en un banco de pruebas ideal para nuevas técnicas de optimización, como el algoritmo de las luciérnagas.

- Aplicación del Algoritmo: El algoritmo de las luciérnagas es adecuado para problemas de optimización complejos y multimodales como el TSP. Este algoritmo puede explorar eficazmente el espacio de soluciones y evitar quedar atrapado en óptimos locales.

Herramientas a Utilizar

- Algoritmo de las Luciérnagas: Este algoritmo bioinspirado se basa en el comportamiento de las luciérnagas para encontrar soluciones óptimas en problemas de optimización. Utilizaremos este algoritmo para explorar el espacio de soluciones del TSP y encontrar rutas óptimas o cercanas al óptimo.
- Lenguaje de Programación: Python será el lenguaje de programación utilizados para implementar el algoritmo. Este lenguaje ofrece bibliotecas y herramientas adecuadas para el desarrollo y la optimización de algoritmos.
- Entorno de Desarrollo: Se utilizarán entornos de desarrollo como Jupyter Notebook para Python para implementar y probar el algoritmo.

Datos a Usar

- Ciudades Colombianas: Se utilizará un conjunto de 10 ciudades colombianas con sus respectivas coordenadas. Estas ciudades fueron seleccionadas considerando su importancia económica y geográfica dentro del país:
  1. Bogotá
  2. Medellín
  3. Cali
  4. Barranquilla
  5. Cartagena
  6. Bucaramanga
  7. Pereira
  8. Santa Marta
  9. Ibagué
  10. Manizales
- Coordenadas: Las coordenadas de las ciudades se listan a continuación.

Estas se obtuvieron utilizando la librería geopy en Python.

$$
\begin{aligned}
&\text{Bogotá} : (4{,}60971, -74{,}08175)\\
&\text{Medellín} : (6{,}25184, -75{,}56359)\\
&\text{Cali} : (3{,}43722, -76{,}5225)\\
&\text{Barranquilla} : (10{,}96854, -74{,}78132)\\
&\text{Cartagena} : (10{,}39972, -75{,}51444)\\
&\text{Bucaramanga} : (7{,}12539, -73{,}1198)\\
&\text{Pereira} : (4{,}81333, -75{,}69611)\\
&\text{Santa Marta} : (11{,}24079, -74{,}20435)\\
&\text{Ibagué} : (4{,}43889, -75{,}23222)\\
&\text{Manizales} : (5{,}06889, -75{,}51738)
\end{aligned}
$$

- Distancias: Las distancias entre las ciudades se calcularon utilizando la fórmula de Haversine.

Contexto: Estado del Arte del Problema del Viajante y el Algoritmo de Luciérnagas

El Problema del Viajante (TSP) continúa siendo un área activa de investigación debido a su complejidad y sus numerosas aplicaciones prácticas. A lo largo de los años, se han desarrollado diversos enfoques para resolver el TSP, desde métodos exactos como la programación lineal entera hasta metaheurísticas como el algoritmo de luciérnagas.

El algoritmo de luciérnagas, una metaheurística inspirada en el comportamiento de las luciérnagas, ha ganado popularidad en las últimas décadas debido a su simplicidad y eficiencia. Investigaciones recientes se han centrado en mejorar el rendimiento del algoritmo mediante la incorporación de nuevas estrategias, como la hibridación con otras metaheurísticas, el ajuste adaptativo de parámetros y la incorporación de mecanismos de aprendizaje automático.

En comparación con otras metaheurísticas, como el recocido simulado y los algoritmos genéticos, el algoritmo de luciérnagas ha demostrado ser competitivo en términos de la calidad de las soluciones encontradas y el tiempo de ejecución. Su capacidad para explorar el espacio de soluciones de manera eficiente y evitar óptimos locales lo convierte en una opción atractiva para resolver problemas de optimización complejos como el TSP.

Las aplicaciones del algoritmo de luciérnagas se han extendido más allá del TSP, abarcando áreas como la optimización de la distribución de planta, la planificación de la producción, la logística, el enrutamiento de vehículos y la optimización de redes. Su versatilidad y adaptabilidad lo convierten en una herramienta valiosa para abordar una amplia gama de problemas de optimización en diferentes campos.

Modelos Matemáticos

Modelo del Problema del Viajante

El Problema del Viajante (TSP) se puede formular matemáticamente como sigue:

Datos:

- Un conjunto de n ciudades, $C = \{c_1, c_2, \ldots, c_n\}$. En este caso, $n = 10$ y cada c_i representa una de las ciudades colombianas seleccionadas.
- Una matriz de distancias D donde D_{ij} representa la distancia entre la ciudad c_i y la ciudad c_j. Los valores de D_{ij} se calcularon utilizando la fórmula de Haversine a partir de las coordenadas de las ciudades.

Objetivo:

Encontrar un ciclo hamiltoniano de mínima longitud en el grafo completo, que visita cada ciudad exactamente una vez y regresa al punto de origen.

Modelo matemático:

El objetivo es minimizar la función de costo:

Minimizar $Z = \sum_{i=1}^{n} \sum_{j=1}^{n} D_{ij} \cdot X_{ij}$

donde:

- x_{ij} es una variable binaria que es 1 si el viajante viaja directamente de la ciudad i a la ciudad j y 0 en el caso contrario.
- **La matriz de distancias D es simétrica, lo que significa que la distancia entre la ciudad i y la ciudad j es igual a la distancia entre la ciudad j y la ciudad i ($D_{ij} = D_{ji}$). Sin embargo, es importante tener en cuenta que el problema del viajante considera la dirección, por lo que el orden en que se visiten las ciudades afecta a la solución final.**

Sujeto a las siguientes restricciones:

- Cada ciudad es visitada exactamente una vez.
- Eliminación de subciclos (para evitar ciclos no deseados).

Modelo de optimización de algoritmo de las luciérnagas

El algoritmo de las luciérnagas no tiene una formulación matemática exacta como el TSP, pero su comportamiento puede modelarse usando la teoría de optimización. En el contexto del TSP, el algoritmo se basa en la siguiente formulación:

Función Objetivo: El objetivo es minimizar la distancia total recorrida. Esto se traduce en maximizar la "brillantez" de las soluciones, donde la brillantez está inversamente relacionada con la distancia total.

Movimiento de las Luciérnagas: Cada luciérnaga se mueve hacia otras luciérnagas con mayor brillantez. La actualización de la posición de una luciérnaga ¡span class=”math-inline»i¡/span¿hacia una luciérnaga ¡span class=”math-inline»j¡/span¿con mejor solución se define como:

$$x_i(t+1) = x_i(t) + \beta e^{-\gamma \cdot d_{ij}} \cdot (x_j(t) - x_i(t)) + \alpha \cdot rand$$

Donde:

- $x_i(t)$ es la posición de la luciérnaga i en el tiempo t. En este contexto, $x_i(t)$ representa una permutación de las ciudades, que define la ruta de la luciérnaga i en la iteración t.
- β es la intensidad de atracción. Un valor alto de β indica una mayor influencia, mientras que un valor bajo implica que las luciérnagas se ven menos afectadas por la posición de las demás.
- γ es el coeficiente de absorción de luz.
- d_{ij} es la distancia entre las luciérnagas i y j. En este caso, d_{ij} se puede interpretar como la diferencia entre la ”brillantez”de las soluciones de las luciérnagas i y j, que está relacionada con la distancia total de las rutas.
- α es el factor de aleatoriedad.
- $rand$ es un término aleatorio.

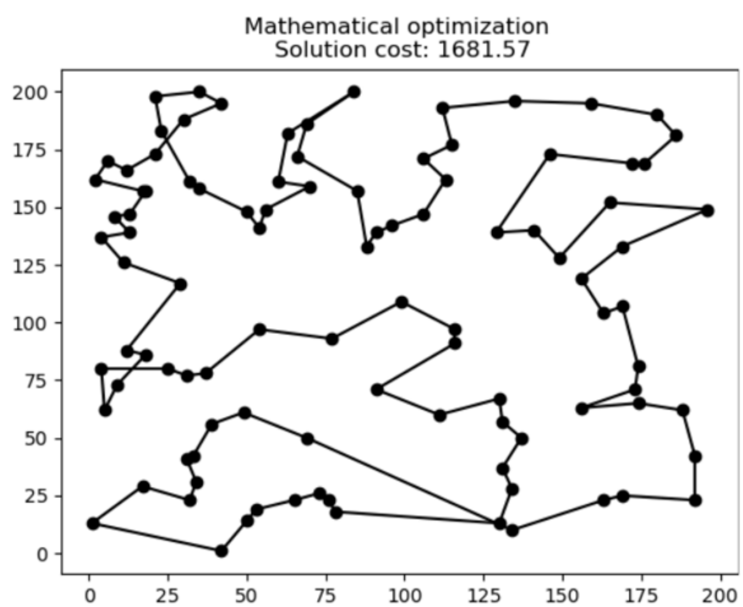


Figura 1: Esquema del movimiento de las luciérnagas en el espacio de soluciones.

El objetivo del algoritmo es encontrar el mínimo o máximo de una función, moviéndose hacia la mejor solución según el brillo relativo de las luciérnagas. El proceso se repite hasta que se alcanza un criterio de parada, como un número máximo de iteraciones o la convergencia a una solución óptima.

Soluciones propuestas:

Programación Lineal Entera:

Como mencionamos, existen dos formulaciones principales para el TSP, el propuesto Miller, Tucker y Zemlin (MTZ) y el Dantzig, Fulkerson y Johnson (DFJ). Aunque la formulación DFJ es más fuerte, la formulación MTZ puede ser útil

Formulación MTZ

La formulación matemática propuesta por Miller, Tucker y Zemlin es la siguiente:

$$\begin{aligned}
I &: \text{set of nodes} \\
x_{ij} &= \begin{cases} 1: & \text{the path goes from node } i \text{ to node } j \\ 0: & \text{otherwise} \end{cases} \\
u_i &: \text{a dummy variable} \\
c_{ij} > 0 &: \text{the distance between node } i \text{ and node } j
\end{aligned}$$

Usando este conjunto, las dos variables y el parámetro, podemos formular el problema como:

Minimizar $Z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ji} x_{ji}$

Sujeto a:

$$\begin{aligned}
& x_{ij} \in \{0,1\} && \forall i,j \in N \\
& u_i \in \mathbb{Z}^+ && \forall i \in N \\
& \sum_{j=1}^{n} x_{ij} = 1 && \forall i \in N \\
& \sum_{i=1}^{n} x_{ij} = 1 && \forall j \in N \\
& \sum_{i=1}^{n} x_{ij} = 1 && \forall j \in N \\
& u_i - u_j + n x_{ij} \leq n - 1 && \forall i,j \in N, i \neq 1, j \neq 1 \\
& 1 \leq u_i \leq n && \forall i \in N, i \neq 1
\end{aligned}$$

La primera y la segunda ecuación refuerzan el tipo de las diferentes variables, la tercera y la cuarta ecuaciones aseguran que cada nodo sea alcanzado y abandonado solo una vez, mientras que las dos últimas ecuaciones imponen que solo una ruta cruce todos los nodos.

Formulación DFJ

En la formulación propuesta por Dantzig, Fulkerson y Johnson partimos de la misma variable x_{ij} y parámetro $c_{ij} > 0$ para que la formulación se pueda completar.

con:

Minimizar $Z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$

Sujeto a:

$$
\begin{aligned}
& x_{ij} \in \{0,1\} && \forall i, j \in N \\
& \sum_{j=1}^{n} x_{ij} = 1 && \forall i \in N \\
& \sum_{i=1}^{n} x_{ij} = 1 && \forall j \in N \\
& \sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1 && \forall S \subset N, S \neq \emptyset
\end{aligned}
$$

Las tres primeras ecuaciones son las mismas que en la formulación anterior, y la nueva restricción (la última) asegura que no haya sub-recorridos, por lo que la solución devuelta es un recorrido único y no la combinación de recorridos disjuntos.

Implementación y Experimentación

Algoritmo de las Luciérnagas en Python

El algoritmo de las luciérnagas se implementó en Python utilizando bibliotecas como NumPy para el manejo de matrices y cálculos numéricos. El código proporcionado muestra la implementación del algoritmo.

Experimentación

Se realizaron experimentos con el conjunto de 10 ciudades colombianas. Se analizó la calidad de la solución obtenida en términos de la longitud de la ruta. Los parámetros utilizados para el algoritmo fueron:

* Número de luciérnagas: 50 * Número de iteraciones: 100 * Factor de atracción: 1

Resultados

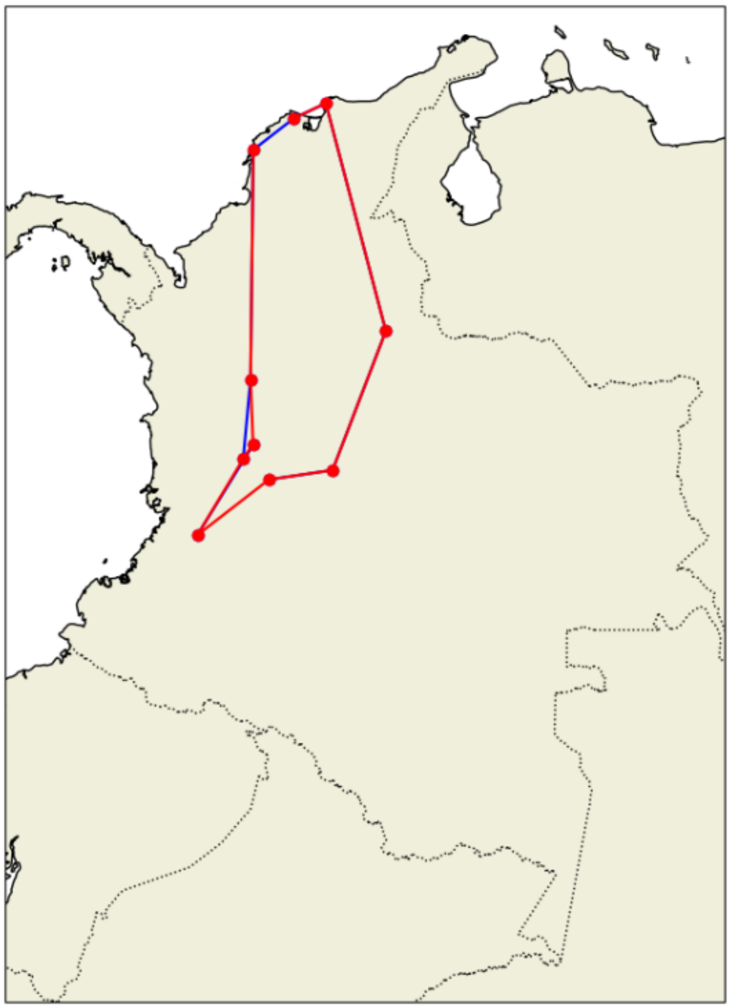
La mejor ruta encontrada por el algoritmo de luciérnagas es la siguiente:

1. Cali 2. Manizales 3. Pereira 4. Medellín 5. Cartagena 6. Barranquilla 7. Santa Marta 8. Bucaramanga 9. Bogotá 10. Ibagué

La distancia total de esta ruta es de 2445.73 km.

Visualización de Resultados

La ruta final obtenida se visualiza en el siguiente mapa:



En este mapa se comparan dos rutas:

Ruta 1:

1. Cali 2. Manizales 3. Pereira 4. Medellín 5. Cartagena 6. Barranquilla 7. Santa Marta 8. Bucaramanga 9. Bogotá 10. Ibagué

Ruta 2:

1. Barranquilla 2. Santa Marta 3. Bucaramanga 4. Bogotá 5. Ibagué 6. Cali 7. Pereira 8. Manizales 9. Medellín 10. Cartagena

La comparación de estas rutas revela que la **Ruta 2 es más eficiente** en términos de distancia total recorrida.

Conclusiones

Este proyecto ha explorado la aplicación del algoritmo de luciérnagas para resolver el Problema del Viajante, un desafío clásico en la optimización combinatoria con amplias aplicaciones en logística y planificación de rutas. A través de la implementación del algoritmo en Python, se ha demostrado su capacidad para encontrar soluciones de buena calidad, es decir, rutas que minimizan la distancia total recorrida, para un conjunto de ciudades colombianas.

La naturaleza bioinspirada del algoritmo de luciérnagas, basada en el comportamiento de estos insectos para encontrar parejas o presas, ofrece una alternativa interesante a los métodos tradicionales de optimización. Su capacidad para explorar el espacio de soluciones de manera eficiente y evitar óptimos locales lo convierte en una herramienta valiosa para abordar problemas complejos como el TSP.

A pesar de su eficiencia, es importante tener en cuenta que el rendimiento del algoritmo puede verse afectado por la elección de los parámetros, como el número de luciérnagas, el factor de atracción y el coeficiente de absorción de luz. La experimentación y el ajuste fino de estos parámetros son cruciales para obtener los mejores resultados.

En conclusión, el algoritmo de luciérnagas se presenta como una técnica prometedora para la resolución del Problema del Viajante. Su capacidad para encontrar soluciones cercanas al óptimo, su flexibilidad y su facilidad de implementación lo convierten en una herramienta valiosa para la optimización de rutas y la planificación logística. Además, el estudio de este algoritmo abre la puerta a la exploración de otras técnicas bioinspiradas para resolver problemas de optimización en diversas áreas. La investigación continua en este campo puede conducir a nuevas soluciones y mejoras en la eficiencia de los algoritmos existentes.

Bibliografía

1. https://www.uaeh.edu.mx/scige/boletin/tlahuelilpan/n3/e5.html
2. https://core.ac.uk/download/pdf/161254599.pdf
3. https://revistas.udistrital.edu.co/index.php/visele/article/view/17474
4. https://ve.scielo.org/scielo.php?script=sci_arttext&pid=S0798-40652015000100010