Problema del Viajante Resuelto con el Bioalgoritmo de las Luciérnagas

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

- Dataset del TSP: Utilizaremos un dataset que contiene un conjunto de ciudades y las distancias entre ellas. Este dataset será proporcionado y subido al repositorio. Los datos estarán en un formato estándar que permitirá su fácil integración con el algoritmo de las luciérnagas.

- Formato del Dataset: El dataset incluirá coordenadas de ciudades o una matriz de distancias. Esto permitirá calcular la ruta más corta posible utilizando el algoritmo de las luciérnagas.

Modelos Matemáticos

Modelo del Problema del Viajante

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

Datos:

- Un conjunto de n ciudades, ¡span class=”math-inline»C^{-}
  {c_1,c_2,
  ldots,c_n
  }¡/span¿

- Una matriz de distancias ¡span class=”math-inline»D¡/span¿donde ¡span class=”math-inline»D_{ij}¡/span¿representan la distancia entre la ciudad ¡span class=”math-inline»c_i¡/span¿y la ciudad ¡span class=”math-inline»c_j¡/span¿

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

donde:

- ¡span class=”math-inline»x_{ij}¡/span¿es una variable binaria que es 1 si el viajante viaja directamente de la ciudad ¡span class=”math-inline»i¡/span¿a la ciudad ¡span class=”math-inline»j¡/span¿y 0 en el caso contrario.

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:

- ¡span class=”math-inline»x_it¡/span¿es la posición de la luciérnaga ¡span class=”math-inline»i¡/span¿en el tiempo ¡span class=”math-inline»t¡/span¿.
- ¡span class=”math-inline»
  beta¡/span¿es la intensidad de atracción.
- ¡span class=”math-inline»
  gamma¡/span¿es el coeficiente de absorción de luz.
- ¡span class=”math-inline»d_{ij}¡/span¿es la distancia entre las luciérnagas ¡span class=”math-inline»i¡/span¿y ¡span class=”math-inline»j¡/span¿.
- ¡span class=”math-inline»
  alpha¡/span¿es el factor de aleatoriedad.
- ¡span class=”math-inline»rand¡/span¿es un término aleatorio.

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:

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

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 ¡span class="math-inline»x_{ij}¡/span¿y parámetro ¡span class="math-inline»c_{ij} 0¡/span¿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 implementará en Python utilizando bibliotecas como NumPy para el manejo de matrices y cálculos numéricos, y Matplotlib para la visualización de resultados.

Experimentación

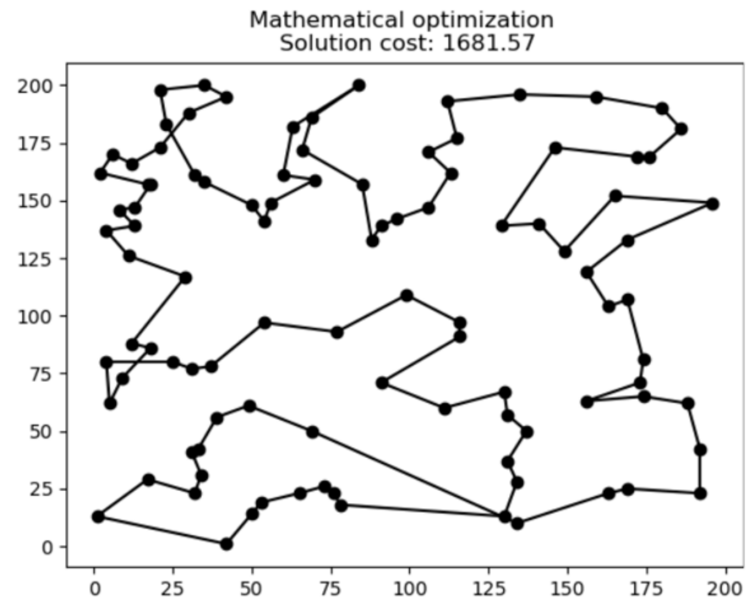
Se realizarán experimentos con diferentes instancias del TSP para evaluar el rendimiento del algoritmo de las luciérnagas. Se analizará la calidad de las soluciones obtenidas en términos de la longitud de la ruta y el tiempo de ejecución.

Resultados Esperados

Se espera que el algoritmo de las luciérnagas encuentre soluciones de buena calidad para el TSP, es decir, rutas cercanas al óptimo. Además, se espera que el algoritmo sea capaz de manejar instancias de diferentes tamaños y complejidades.

Visualización de Resultados

Los resultados se visualizarán mediante gráficos que muestren la evolución de la mejor solución encontrada a lo largo de las iteraciones del algoritmo. También se mostrará la ruta final obtenida en un mapa o gráfico.



Conclusiones

Se espera que este proyecto demuestre la efectividad del algoritmo de las luciérnagas para resolver el Problema del Viajante. Los resultados obtenidos proporcionarán información valiosa sobre el rendimiento del algoritmo y su aplicabilidad en problemas de optimización de rutas.