Reporte de Optimización

Materia: Optimización

Alumnos

- Semíramis García de la Cruz
- Federico Salinas Samaniego

Traveling Salesman Problem

Descripción del problema

El problema del agente viajero – Travel Salesman Problem (TSP por sus siglas en inglés) es un problema clásico de optimización con diversas aplicaciones en el mundo real donde se busca encontrar la ruta más eficiente para que un viajero visite un conjunto de ciudades y regrese a su punto de origen, minimizando la distancia total recorrida.

Matemáticamente, el TSP puede ser descrito con los siguientes componentes.

Parámetros

1. **Conjunto de nodos:** $N = \{1, 2, ..., n\}$ un conjunto de ciudades que se deben visitar.

2. **Matriz de distancias:** $D = [d_{ij}]_{n \times n}$ una matriz $n \times n$, donde d_{ij} representa la distancia entre la ciudad *i* y la ciudad *j*.

Variables de decisión

Serán de la forma x_{ij}, variables binarias, que serán 1 si el camino entre las ciudades *i* y *j* se incluye en la solución y 0 en caso contrario.

Función objetivo

Para el TSP, se busca la minimización de la distancia total recorrida, que puede representarse como

$$min \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_{ij}$$

Factibilidad

Las restricciones del problema son las siguientes:

- Cada nodo es visitado exactamente una vez

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1, \forall i \in N$$

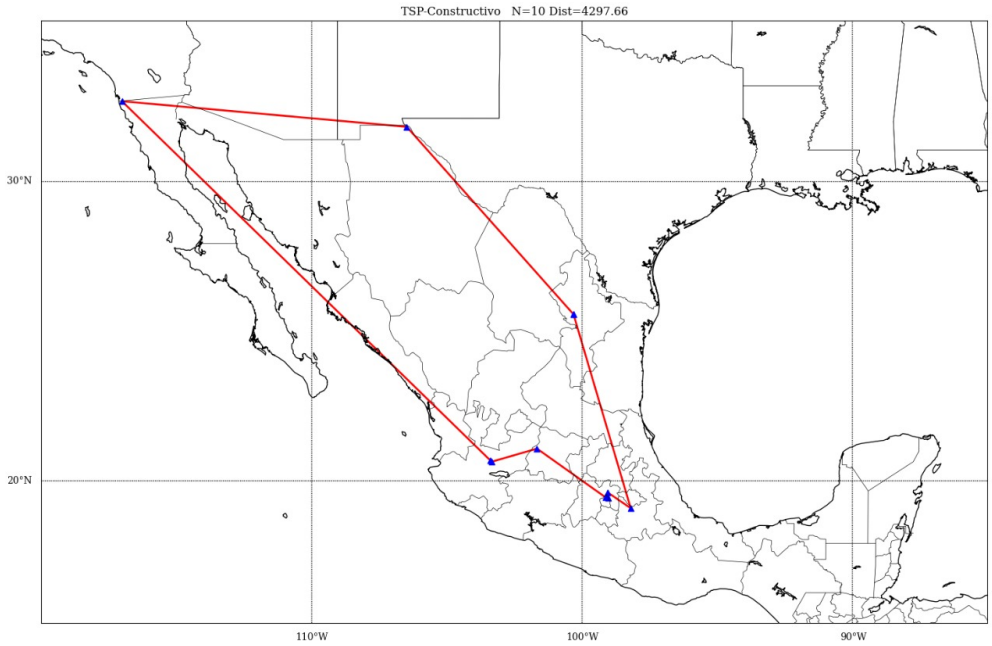
- El agente regresa a la ciudad de origen

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1, \forall j \in N$$

- No existen sub-tours, es decir, viajes que no incluyan todas las ciudades.

$$u_i - u_j + n x_{ij} \leq n - 1, \forall i, j \in N, i \neq j, j > 1, u_1 = 1$$

  Donde u_i es una variable que indica el orden en el cual se visitan las ciudades. Esta formulación garantizada que el agente recorra todas las ciudades exactamente una vez y regrese al punto de partida, sin formar sub-tours.



Ejemplo de implementación del método TSP constructivo aleatorio, en las capitales de México. Creación propia.

Supongamos que un vendedor necesita visitar 4 ciudades: A, B, C y D. Las distancias entre las ciudades son las siguientes:

	A	B	C	D
A	0	10	15	20
B	10	0	35	25
C	15	35	0	30
D	20	25	30	0

La ruta más corta posible para el vendedor en este ejemplo es A → B → D → C → A, con una distancia total de 80 unidades.

Al realizar la búsqueda de soluciones completamente aleatorias, la implementación en Python regresa la siguiente solución

```
-- Random Constructive Method --
N = 4
Initial Node: A

Searching solutions...
#1 - 95.0
#3 - 80.0
['A', 'B', 'D', 'C', 'A'] 80.0
```

El orden de la secuencia obtenida, así como la distancia recorrida asociada, es la misma que se reporta. Por ende, la implementación es capaz de hallar buenas soluciones para el TSP.

Descripción del método constructivo del problema

1. Iteración:
    - Mientras el número de iteraciones realizadas (`n_iters`) sea menor que el máximo permitido (`MaxIters`).
2. Generación de Ruta Aleatoria:

- Calcular el número total de nuevas selecciones de nodos (`TotalNewSelections`) como la cantidad total de nodos menos uno.
- Inicializar una ruta (`route`) con el nodo inicial (`Ini`) al principio y al final, y marcando el resto como "*".
- Seleccionar índices aleatorios de nodos disponibles (`random_indexes`) sin reemplazo.
- Concatenar los índices de nodo seleccionados con el nodo inicial para formar una lista de índices de distancia (`dist_indices`).
- Asignar los nombres correspondientes a los nodos seleccionados en la ruta.
- Calcular la distancia recorrida (`dist_rec`) sumando las distancias entre nodos consecutivos en `dist_indices`.

3. Evaluación de la Solución Actual:
   - Incrementar el contador de iteraciones (`n_iters`) en uno.
   - Si la distancia recorrida (`dist_rec`) es menor que la distancia mínima encontrada hasta el momento (`min_dist`):
     - Actualizar la mejor ruta (`best_route`) con la ruta actual.
     - Actualizar la distancia mínima (`min_dist`) con la distancia recorrida.
     - Imprimir el número de iteración actual (`n_iters`) junto con la distancia mínima encontrada (`min_dist`).