Universidad de Concepción
Facultad de Ingeniería
Departamento de Informática y Ciencias de la Computación

Tarea Computacional 2

El problema del Vendedor Viajero y su solución mediante Python con PuLP

Integrantes: Roberto Felipe Artigues Escobar
Emilio Juan Meza Quiroz

Profesora: Rosa Medina

Fecha de entrega: 10 de Noviembre de 2023
Concepción, Chile

Situación Propuesta

En el año 2150, la humanidad ha establecido colonias en diversos sistemas estelares. La empresa Çourier Cósmico S.A."se enfrenta al desafío de optimizar las rutas de sus naves espaciales para el abastecimiento de estas colonias. El costo de viajar entre colonias es asimétrico debido a factores como las diferencias gravitacionales, corrientes espaciales y eventos cósmicos. Este problema se conoce como el "Problema del Viajante Espacial"(PVE).

El objetivo es diseñar una ruta que permita a una nave visitar cada colonia una sola vez y regresar a la base terrestre de la manera más eficiente posible. La nave debe llevar suministros vitales, realizar mantenimientos y transportar pasajeros, todo esto bajo la restricción de minimizar los costos totales que incluyen consumo de combustible, tiempo, peajes espaciales y maniobras de asistencia gravitatoria.

1. Plantamiento del problema

En el contexto de la Optimización de Rutas Espaciales Intercoloniales (OREI), consideramos el conjunto de colonias interstelares que deben ser visitadas exactamente una vez por una nave espacial, que parte y regresa a la base terrestre. El costo c_{ij} representa el consumo de combustible, tiempo, peajes espaciales y maniobras gravitatorias entre la colonia i y la colonia j, y no es simétrico debido a las diferentes condiciones espaciales.

Variables de Decisión

Las variables de decisión definen las opciones operativas que el modelo puede tomar para resolver el problema. Estas son:

- x_{ij}: Variable binaria que indica si la nave realiza un viaje directo de la colonia i a la colonia j. Toma un valor de 1 si la ruta se elige y 0 en caso contrario.
- u_i: (Solo en la formulación MTZ) Variable continua que representa el orden de visita de la nave a la colonia i, ayudando a evitar subciclos en la ruta.
- g_{ij}: (Solo en la formulación GG) Variables de flujo que indican el número de naves que viajan entre las colonias i y j, asegurando la conservación del flujo y evitando la formación de subrutas.

Parámetros del Problema

Los parámetros del problema son fundamentales para definir el contexto y las limitaciones del modelo. Estos incluyen:

- n: Número total de colonias interstelares a considerar en el modelo.
- c_{ij}: Costo asociado con el viaje de la nave espacial desde la colonia i hasta la colonia j. Este costo es asimétrico y comprende factores como el consumo de combustible, tiempo de viaje, peajes espaciales, y maniobras de asistencia gravitatoria.

Formulación de Dantzig-Fulkerson-Johnson (DFJ)

Función Objetivo:

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

Sujeto a:

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad \text{para toda colonia } j \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad \text{para toda colonia } i \tag{3}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \qquad \text{para cualquier subconjunto } S \text{ de colonias, con } S \neq \emptyset \tag{4}$$

$$x_{ij} \in \{0, 1\}, \qquad \text{si la nave viaja de } i \text{ a } j \tag{5}$$

Formulación de Miller-Tucker-Zemlin (MTZ)

Incluye todas las restricciones de la formulación DFJ y añade:

$$u_i - u_j + (n-1)x_{ij} \leq n - 2, \quad \text{para } i, j = 2, \ldots, n \tag{6}$$

Donde u_i es una variable continua que representa el orden de visita a la colonia i en la ruta, ayudando a prevenir subrutas dentro del recorrido.

Formulación de Gillett-Gomory (GG)

Incluye todas las restricciones de la formulación DFJ y añade:

$$\sum_{j=1}^{n} g_{ij} - \sum_{j=2}^{n} g_{ji} = 1, \qquad \text{para toda colonia } i \text{ excepto la base terrestre} \tag{7}$$

$$0 \leq g_{ij} \leq (n-1)x_{ij}, \qquad \text{si hay flujo de la nave entre } i \text{ y } j \tag{8}$$

donde g_{ij} son variables de flujo que representan el número de naves entre las colonias i y j, asegurando la conservación del flujo y la no formación de subrutas.

2. Análisis de Resultados

Utilizamos un conjunto de 10 instancias de matrices de distancia asimétricas generadas mediante el archivo `atsp_generator.py` (código adjunto en el archivo ZIP). Este conjunto consta de 4 instancias de tamaño 10 (1, 2, 3 y 4), 3 instancias de tamaño 13 (5, 6 y 7), y 3 instancias de tamaño 16 (8, 9 y 10). Posteriormente, aplicamos diversos algoritmos en las 10 instancias, realizando en promedio 15 repeticiones por cada instancia. A continuación se presetan dos tablas, la primera con los resultados óptimos y la segunda con los tiempos de computo por cada instancia:

	1	2	3	4	5	6	7	8	9	10
DFJ	38	45	27	49	43	41	41	41	40	46
MTZ	38	45	27	49	43	41	41	41	40	46
GG	38	45	27	49	43	41	41	41	40	46

Tabla 1: Resultados Óptimos

Instancia	DFJ (s)	MTZ (s)	GG (s)
Inst. 1	0.115	0.067	0.069
Inst. 2	0.287	0.126	0.143
Inst. 3	0.124	0.058	0.062
Inst. 4	0.114	0.043	0.055
Inst. 5	1.298	0.171	0.215
Inst. 6	7.264	0.124	0.233
Inst. 7	1.279	0.119	0.083
Inst. 8	15.691	0.181	0.153
Inst. 9	16.221	0.157	0.190
Inst. 10	29.835	0.659	0.702

Tabla 2: Tiempos de Cómputo por Instancia y Modelo

En cuanto a los resultados obtenidos, se puede observar en la tabla adjunta que el algoritmo DFJ presenta un rendimiento variado en las diferentes instancias de tamaño proporcionadas. En particular, destaca en las instancias 5, 6, 8 y 9, donde muestra un aumento significativo en el tiempo de cómputo, alcanzando valores notables de 1.30, 7.26, 15.69 y 16.22 segundos, respectivamente. Este comportamiento sugiere que la implementación del algoritmo DFJ podría estar experimentando desafíos en instancias más complejas.

Por otro lado, los algoritmos MTZ y GG exhiben tiempos de cómputo más consistentes a lo largo de todas las instancias. En general, MTZ se destaca como el algoritmo más eficiente

en términos de tiempo de ejecución, manteniendo tiempos relativamente bajos incluso en las instancias más desafiantes, como se observa en las instancias 5, 6, 8 y 9.

En resumen, mientras DFJ presenta una variabilidad en su desempeño, MTZ emerge como el algoritmo más eficaz en términos de tiempo de cómputo, con GG mostrando eficiencia comparable en la mayoría de las pruebas y superándolo en la instancia 8.

Conclusión

En este trabajo, hemos abordado el "Problema del Viajante Espacial"(PVE), un desafío complejo y significativo en el contexto de la optimización de rutas intercoloniales en el año 2150. Mediante el uso de diferentes algoritmos, como DFJ, MTZ y GG, hemos logrado identificar rutas óptimas y eficientes para el abastecimiento de las colonias espaciales, considerando una variedad de factores como los costos de combustible, tiempo, y peajes espaciales.

Los resultados indican que, aunque el algoritmo DFJ muestra variabilidad en su rendimiento, especialmente en instancias de mayor complejidad, los algoritmos MTZ y GG ofrecen un desempeño más consistente y eficiente. En particular, MTZ se destaca por su eficacia en términos de tiempo de cómputo, incluso frente a instancias más desafiantes, lo que sugiere su viabilidad para aplicaciones prácticas en la planificación de rutas espaciales.

Estos hallazgos abren el camino para futuras investigaciones, donde se podría explorar la adaptación y mejora de estos algoritmos para afrontar desafíos aún más complejos en la logística espacial. Asimismo, se recomienda examinar el impacto de otros factores, como las condiciones dinámicas del espacio y la variabilidad en la demanda de las colonias, para seguir optimizando las rutas y contribuir al desarrollo sostenible de las colonias interstelares.