Facultad de Ingeniería

Optimización

PROYECTO OPTIMIZACIÓN

Los Tres Problemas NP-Hard

Autores:

Jairo Andrés Carrasco Gárate

Fabián Nicolás Castillo Teran

Hans Kevin Chicaiza Barbosa

Luis Alfredo Flores Arriagada

Profesor Guía:

Jorge Humberto Sabattin Ortega

Santiago, Chile

2024

Índice:

Problemas de Ruteo de Vehículos (VRP)

Los Problemas de Ruteo de Vehículos (VRP) son una clase de problemas de optimización que buscan determinar las rutas óptimas para una flota de vehículos que deben visitar un conjunto de clientes o ubicaciones.

Estos problemas son fundamentales en logística y gestión, y hay muchas variantes específicas del problema que modifican las restricciones, haciendo el problema más complejo.

Open Vehicle Routing Problem (OVRP)

En este problema, los vehículos no están obligados a regresar al depósito después de completar sus entregas.

Este problema resulta útil en situaciones donde los vehículos pueden terminar su ruta en una ubicación diferente a la de inicio.

Características Clave:

- Los vehículos inician en un depósito.
- No hay obligación de regresar al depósito al finalizar la ruta.

Desafíos Computacionales:

La principal complejidad radica en encontrar la ruta óptima sin el requisito de retorno, lo que puede aumentar el espacio de búsqueda y la dificultad de optimización.

A continuación se presenta el modelo matemático de OVRP mostrado en https://repositorio.utp.edu.co/server/api/core/bitstreams/3e902645-43b0-4a32-b898-fcf71d2c52a0/content

Modelo Matemático:

Variables de decisión:

$$S_{ij} = \begin{cases} 1 \text{ cuando el arco } i \text{ a } j \text{ es recorrido por un vehiculo} \\ 0 \text{ cuando el arco } i \text{ a } j \text{ no es recorrido por un vehiculo} \end{cases}$$

$L_{ij} = Acumulado\ de\ carga\ en\ el\ nodo\ i\ del\ vehículo\ j$

$I_{ij} = carga\ \ en\ el\ nodo\ i\ del\ vehículo\ j$

Parámetros:

$C_{ij} = matriz\ de\ distancias\ entre\ nodos\ y\ entre\ depósito\ y\ nodos$

$D_j = demanda\ de\ cada\ nodo$

$Q = capacidad\ del\ vehículo$

$C_x = coordenadas\ del\ cliente\ y\ depósito\ en\ el\ eje\ x$

$C_y = coordenadas\ del\ cliente\ y\ el\ depósito\ en\ el\ eje\ y$

Función objetivo:

$$\sum_{i \in V} \sum_{j \in V} C_{ij} S_{ij} \qquad (3.2)$$

Sujeto a:

$$\sum_{i \in V} S_{ij} = 1 \quad para\ j \in J\ y\ todo\ i \neq j \qquad (3.3)$$

$$\sum_{k \in J} S_{jk} \leq \sum_{i \in V} S_{ij} \ \ para\ todo\ j \in J\ y\ todo\ j \neq k\ y\ i \neq j \qquad (3.4)$$

$$S_{ij} + S_{ji} \leq 1\ para\ todo\ i \in V\ y\ j \in V \qquad (3.5)$$

$$L_{ij} \leq Q S_{ij} \ \ para\ todo\ i \in V\ y\ j \in V \qquad (3.6)$$

$$\sum_{i \in I} \sum_{j \in J} S_{ij} \geq \frac{\sum_{i \in J} D_j}{Q} \qquad (3.7)$$

$$\sum_{i \in V} S_{ij} \geq \sum_{k \in J} S_{jk} \quad para\ todo\ j \in J \qquad (3.8)$$

$$\sum_{i \in V} I_{ij} = \sum_{k \in V} I_{jk} + D_j \ \ para\ todo\ j \in J, todo\ i \neq j\ y\ k \neq j \qquad (3.9)$$

$$S_{ij} \ \ binaria \qquad (3.10)$$

$$L_{ij} \geq 0 \ \ y \ \ L_{ij} \leq Q \qquad (3.11)$$

Imagen 1: Variables, Parámetros y Modelo Matemático

Función Objetivo:

Minimizar el costo de la ruta y lleva implícita la minimización del número de vehículos.

Restricciones:

(3.3). Cada nodo tendrá un único arco de llegada.

(3.4). Permite asegurar que un nodo tendrá el mismo arco de entrada como de salida.

(3.5). Esta restricción evita la duplicación de arcos e indica el sentido del mismo.

(3.6). Con esta restricción se garantiza el balance de flujos de demanda en los vehículos. Se tiene la variable como la cantidad de carga que lleva un vehículo. Esta restricción indica que el flujo de carga que sale de un nodo es igual al flujo que llega más la demanda del mismo nodo.
(3.7). Se garantiza que el número de rutas es suficiente para atender toda la demanda de los clientes. Esto se consigue haciendo que la sumatoria de todos los arcos que salen del depósito sean igual (o eventualmente menor si la demanda puede ser atendida con menos vehículos) al cociente de la demanda total con la capacidad de los vehículos.
(3.8). Garantiza la continuidad de la ruta.
(3.9). Están son las encargadas del balance de las rutas, es decir garantizan que lo que entre a un nodo sea igual a lo que sale más la demanda o lo que se quedó en ese nodo.

Implementación Algoritmo:

Para abordar este problema, evaluamos un código en Python que presenta una simplificación del escenario del Problema OVRP. Este código se centra en una matriz de dimensiones 10x10, donde cada elemento representa la distancia entre nodos, que en este contexto, son clientes.

Enlace de código(OVRP VS): https://github.com/BluSwing/Proyecto-Optimizaci-n

Nuestro primer código implementa la heurística Nearest Neighbor, un enfoque que busca construir una ruta iterativamente seleccionando el nodo más cercano al último nodo agregado. Esta estrategia resultó en un tiempo de ejecución sorprendentemente bajo, encontrando una distancia mínima de 37 unidades como se puede observar en la imagen 2.

```
Mejor ruta encontrada: [0, 1, 2, 3, 5, 9, 8, 4, 6, 7]
Distancia mínima: 37
Tiempo de ejecución: 0.000026 segundos
```

Imagen 2: Resultado código creado por nosotros del problema OVRP.

Para el segundo código creado por nosotros fue implementado en python modelado como AMPL con Gurobi con esta formulación:

- Variables: Se definen variables para representar las rutas de los vehículos y las conexiones entre los clientes y el depósito. En el OVRP, se pueden usar variables binarias para indicar si se visita un cliente o no.

- Función Objetivo: Generalmente se busca minimizar los costos totales, como la distancia recorrida o el tiempo de viaje de los vehículos.

- Restricciones: Además de las restricciones de capacidad de los vehículos y la visita única a los clientes, el OVRP puede incluir restricciones adicionales como limitaciones en el número de vehículos disponibles o restricciones de tiempo de viaje total.

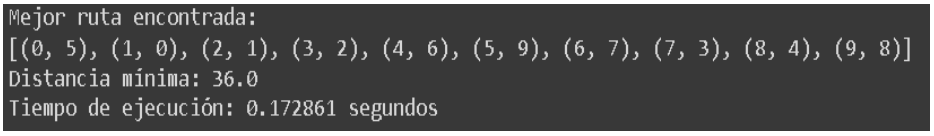

Imagen 3: Resultado código creado por nosotros del problema OVRP Gurobi

El código generado por ChatGPT no emplea una heurística específica, sino que implementa un enfoque de fuerza bruta para resolver el problema. Este método evalúa exhaustivamente todas las posibles rutas para encontrar la distancia mínima. Aunque el tiempo de ejecución es significativamente mayor en comparación con nuestro código, que utiliza la heurística Nearest Neighbor, logró encontrar una ruta con una distancia mínima de 36 unidades, una unidad mejor que la encontrada por nuestro primer código, que tenía una distancia mínima de 37. Sin embargo, nuestro segundo código también encontró una distancia mínima de 36 unidades, al igual que el de ChatGPT, aunque con un tiempo de ejecución más rápido, demostrando ser más eficiente.

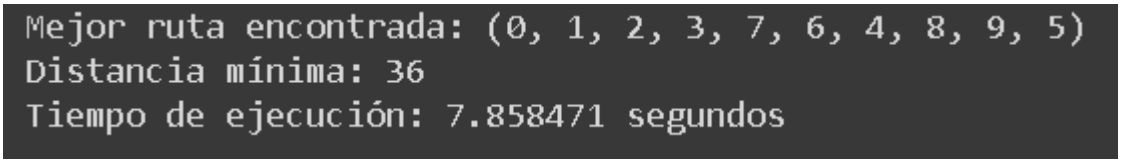

Imagen 4: Resultado código creado por ChatGPT del problema OVRP.

Capacitated Vehicle Routing Problem (CVRP)

En este problema, cada vehículo tiene una capacidad máxima que no puede ser excedida por la demanda total de los clientes en su ruta, que tiene por objetivo minimizar la distancia total recorrida, asegurando que la capacidad de los vehículos no se sobrepase.

Alguna de las características clave::

- **Vehículos con Capacidad Limitada:** Cada vehículo en la flota tiene una capacidad máxima de carga que no puede ser excedida y dicha capacidad debe ser suficiente para manejar la demanda de los clientes asignados a su ruta.
- **Demanda de los Clientes:** Cada cliente tiene una demanda específica que debe ser satisfecha.
  La demanda puede ser en términos de volumen, peso u otra métrica relevante para la carga.
- **Un Solo Depósito:** Los vehículos inician y terminan su ruta en un depósito central y todas las rutas deben comenzar y finalizar en este depósito.
- **Objetivo de Optimización:** El objetivo principal es minimizar el costo total de las rutas, que generalmente se mide en términos de la distancia recorrida, el tiempo de viaje o el costo de transporte.
  Alternativamente, puede haber otros objetivos, como minimizar el número de vehículos utilizados o equilibrar la carga entre los vehículos.
- **Restricciones de Rutas:** Cada cliente debe ser visitado exactamente una vez por un solo vehículo. La suma de las demandas de los clientes en una ruta no debe exceder la capacidad del vehículo asignado.

Desafíos Computacionales:

La capacidad de los vehículos y su restricción con la ruta y los nodos que visita el vehículo añaden una capa adicional de restricciones al problema original VRP que complica la optimización y requiere técnicas avanzadas para encontrar soluciones factibles y óptimas.

Modelo Matemático:

A continuación se presenta el modelo matemático de CVRP mostrado en (PDF) Vehicle routing problem: Models and solutions que fue una representación de la formulación hecha por (Fukasawa et al. 2004)

Definiciones previas:

H = (N, A): Representa un grafo donde N es el conjunto de vértices y A el conjunto de aristas.

d(e): Representa la distancia o costo asociado a la arista

q: La demanda de cada cliente.

Q: La capacidad de cada vehículo.

El vértice 0 es el depósito.

x sub(e): Variable binaria que indica si la arista **e** es utilizada en la solución.

λ: Variables auxiliares utilizadas para asegurar la conectividad y la satisfacción de la demanda.

$$\text{Minimise} \sum_{e=(u,v)\in A} d(e)\, x_e \qquad (25)$$

Subject to:

$$\sum_{e\in\delta(\{u\})} x_e = 2 \ , \forall\ u\in\ N\backslash\{0\}, \qquad (26)$$

$$\sum_{e\in\delta(\{0\})} x_e \geq 2\ k^* , \qquad (27)$$

$$\sum_{e\in\delta(S)} x_e \geq 2\, k(S) , \forall\ S \in N\backslash\{0\}, \qquad (28)$$

$$x_e \leq 1 , \ \forall e\in A\backslash\delta(\{0\}) , \qquad (29)$$

$$\sum_{l=1}^{p} q_l^e\, \lambda_l - x_e = 0 , \forall e\in A , \qquad (30)$$

$$x_e \in\{0,1,2\} , \quad \forall e\in A , \qquad (31)$$

$$\lambda_l \geq 0 \quad \forall\ l \in \{1,\ldots, p\}. \qquad (32)$$

Imagen 5: Modelo Matemático CVPR

Función objetivo (25): busca minimizar el costo total de las rutas, donde el costo está dado por la suma de las distancias **d(e)** de las aristas seleccionadas.

Restricción (26): Cada cliente **u** debe ser visitado exactamente una vez, entrando y saliendo (dos aristas por cada cliente).

Restricción (27): El depósito debe ser el punto de inicio y fin de las rutas, con al menos **2k*** aristas, donde **k*** es el número mínimo de vehículos necesarios.

Restricción (28): Para cualquier subconjunto **S** de clientes, debe haber al menos **2k(S)** aristas conectándolos, donde **k(S)** es el número de rutas necesarias para servir a los clientes en **S**.

Restricción (29): Las aristas pueden ser seleccionadas como máximo una vez.

Restricción (30): Asegura que el flujo de vehículos sea coherente con la selección de aristas y la demanda de los clientes.
Restricción (31): Las variables son binarias, indicando si una arista es utilizada o no.
Restricción (32): No Negatividad de Variables Auxiliares

Enlace de código(CVRP VS): https://github.com/BluSwing/Proyecto-Optimizaci-n

Para la compresión de este problema comparamos un código en el lenguaje de programación Python, que tiene una escena simple del problema, el cual consta de una matriz 16x16 en el cual cada elemento representa la distancia entre el centro de distribución (0) y los 16 clientes que deben abastecer.
Problema de Enrutamiento de Vehículos Capacitados (CVRP) con Google OR-Tools
Simple, pero nos permite una base para entender instancias más complejas de este problema y lograr hacer una comparativa con una propuesta de solución realizada por ChatGPT.
El primer código usó una heurística y una metaheurística para resolver el problema, los cuales son:
Estrategia de Primera Solución (First Solution Strategy)

- Es una heurística que se usa para generar rápidamente una solución inicial.
- Selecciona el arco (camino) más barato disponible en términos de distancia o costo.
- Esta estrategia construye la ruta inicial de manera rápida y sencilla, eligiendo siempre el siguiente paso más barato.

Metaheurística de Búsqueda Local Guiada (Guided Local Search)

- Es una técnica más avanzada que mejora la solución inicial obtenida.
- Modifica la solución inicial iterativamente, explorando soluciones vecinas para encontrar una de menor costo.
- Penaliza ciertos arcos en la solución inicial para diversificar la búsqueda y evitar óptimos locales, buscando una solución más cercana al óptimo global.

Estrategia de Segunda Solución (Gurobi modelado como AMPL en Python)

- Definición de variables de decisión que representen las rutas de los vehículos y la asignación de clientes a estas rutas.
- Establecimiento de la función objetivo que generalmente busca minimizar los costos totales, como la distancia recorrida o el tiempo de viaje.
- Formulación de restricciones para garantizar que las rutas respeten la capacidad de los vehículos y que todos los clientes sean atendidos exactamente una vez.

Mientras que la estrategia usada en el código generado por ChatGPT fueron:

Heurística de Ahorro (Savings Heuristic): Calcula los ahorros para cada par de nodos. El ahorro se define como la reducción en la distancia total cuando dos nodos son visitados consecutivamente en lugar de regresar

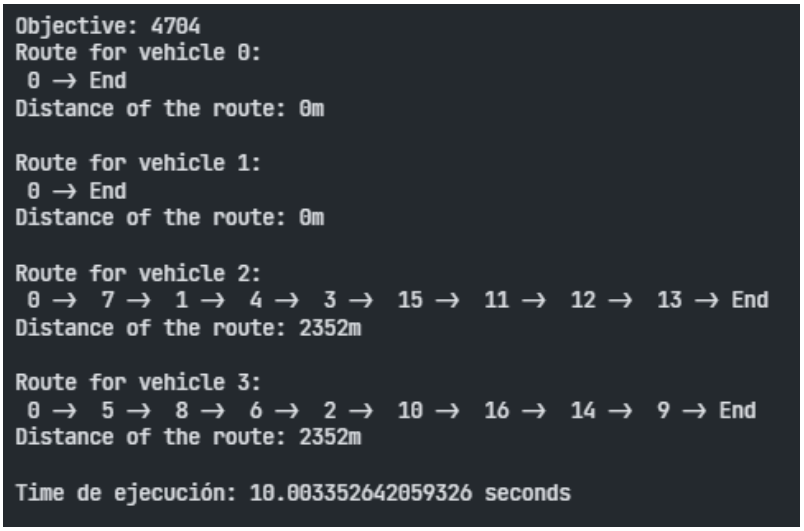

Imagen 6: Resultado código creado por Google Or-tools del problema CVRP.

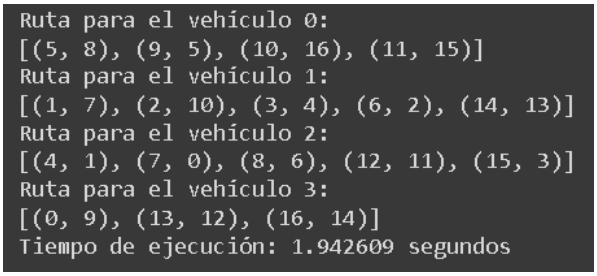

Imagen 7: Resultado código creado por nosotros del problema CVRP.

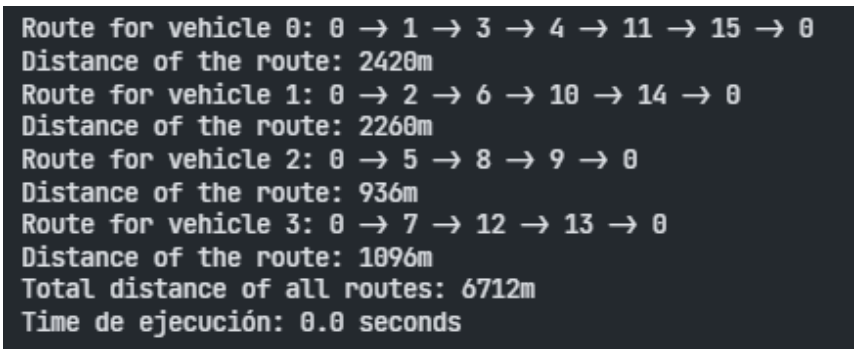

Imagen 8: Resultado código creado por ChatGPT del problema CVRP.

En las imágenes anteriores vemos que el primer código usa:
2 vehículos con una distancia total de 4704 y tomo 10 segundo para encontrar dicha solución, mientras que el código creado en gurobi por nosotros y ChatGPT uso 4 vehículos con una distancia total de 6712 y aproximadamente 0 segundos para encontrar dicha solución en el caso de chatGPT y casi 2 segundos el creado por nosotros.

Estas diferencias resaltan la importancia de la estrategia implementada y los criterios de optimización elegidos, asimismo la diferencia de tiempo, puede ser atribuido a diferencias en la infraestructura de ejecución y capacidades de procesamiento.
En este caso depende qué criterios de optimización escogidos son más óptimos para el problema, aunque generalmente será el costo en dinero que se puede sacar a partir de la distancia total recorrida por los vehículos en cuyo caso sería el primer código mejor, pero si consideramos más importante la rapidez con que se encuentre la solución ChatGPT se adaptó mejor a dicho problema.

Teniendo estos 3 criterios que sería tiempo, vehículos, distancia, el algoritmo que satisface más los requerimientos del problema es el primer código, siendo este más opción en ⅔ criterios, siendo superado en cuanto a tiempo por el generado por ChatGPT.

Vehicle Routing Problem with Time Windows (VRPTW)

En este problema cada cliente tiene un intervalo de tiempo (ventana de tiempo) dentro del cual debe ser visitado.

Los vehículos deben planificar sus rutas no solo considerando las distancias y capacidades, sino también respetando estas restricciones de tiempo.

Características Clave:

- Cada cliente tiene una ventana de tiempo específica para ser atendido.
- Los vehículos deben planificar sus rutas para cumplir con las ventanas de tiempo.
- Puede incluir restricciones de capacidad.

Desafíos Computacionales:

Las ventanas de tiempo añaden una dimensión adicional de complejidad, ya que no solo se debe optimizar la distancia y la capacidad, sino también coordinar los tiempos de llegada dentro de los intervalos permitidos.

Variables:

- x_{ij}: Variable binaria que indica si se realiza la ruta directa desde el nodo i al nodo j.
- t_i: Tiempo de llegada al nodo i.
- q_i: Cantidad de carga en el vehículo al llegar al nodo i.

Parámetros:

- n: Número de nodos (incluyendo el depósito y los clientes).
- q: Capacidad máxima de carga de cada vehículo.
- τi : Tiempo de servicio en el nodo i.
- δi : Ventana de tiempo (inicio).
- Δi Ventana de tiempo (fin).
- d_i: Demanda del cliente en el nodo i.
- M: Número máximo de vehículos disponibles.

Función Objetivo:

$\text{Min} \sum_{i \in N} \sum_{i \in N} c_{ij} x_{ij}$

Donde c_{ij} es el costo (distancia) de viajar del nodo i al nodo j, y N es el conjunto de todos los nodos.

Restricciones

1.Visita a todos los clientes:

$\sum_{j \in N,\, j \neq i} x_{ij} = 1, \forall i \in C$, donde C es el conjunto de clientes

2.Depósito como inicio:

$\sum_{j \in N,\, j \neq 1} x_{1j} = M$

$\sum_{j \in N,\, j \neq 1} x_{i11} = M$

3.Capacidad del vehículo:

$\sum_{i \in N} q_i x_{ij} \leq q \; \forall j \in N$

4.Ventanas de tiempo

$\Delta_i \leq t_i \leq \Delta_i, \; \forall i \in C$

5.Relación de Tiempo de llegada y servicio:

$t_j \leq t_i + \tau i - M(1 - x_{ij}), \; \forall i, j \in N, i \neq j$

Imagen 9: Modelo Matemático y restricciones VRPTW

Función Objetivo

Minimizar distancia recorrida y con esto optimizar las rutas de los vehículos

Restricciones

1. **Visita a todos los clientes:** Cada cliente debe ser visitado exactamente una vez por exactamente un vehículo.
2. **Depósito como inicio y fin:** Cada vehículo debe comenzar y terminar su ruta en el depósito.
3. **Capacidad del vehículo:** La capacidad total transportada por cada vehículo no debe exceder su capacidad máxima.
4. **Ventanas de tiempo:** Cada cliente debe ser visitado dentro de su ventana de tiempo especificada.
5. **Relación tiempo de llegada y servicio:** El tiempo de llegada a cada cliente debe ser mayor o igual al tiempo de llegada al cliente anterior más el tiempo de servicio en ese cliente.

Implementación Algoritmo:

- El conjunto de prueba que se usó en el algoritmo fue el c101 de Solomon

- Todas las constantes son números enteros.
- El cliente número 0 indica el depósito, donde todos los vehículos deben comenzar y terminar.
- K es el número máximo de vehículos.
- Q la capacidad de cada vehículo.
- El 'tiempo de preparación' es el momento más temprano en el que el servicio puede comenzar en un cliente/depósito determinado.
- La 'fecha de vencimiento' es la última hora en la que el servicio puede comenzar en el cliente/depósito determinado.
- El valor del tiempo de viaje es igual al valor de la distancia.

Enlace de código(VRPTW_gurobi, VRPTW-ga-master.zip, Github VRPTW Nosotros(LS).txt): https://github.com/BluSwing/Proyecto-Optimizaci-n

```
Distancia Total 10383.437763863496
Tiempo de Ejecución: 2.6696341037750244 segundos
```

Imagen 10: Resultado código creado por nosotros del problema VRPTW(ILS)

```
Solution count 0
```

Imagen 11: Resultado código creado por nosotros del problema VRPTW con Gurobi

```
value= 17745.035679629145
```

```
Enter run file name [default: C101_200]:
Running: "C101_200"
--- distance table pre-computation time: 0:00:00.002525
### Process time of 50 generation: 0:00:00
### Process time of 50 generation: 0:00:01.464719
```

Imagen 12: Resultado código creado por ChatGPT del problema VRPTW(Genético)

Antes de comenzar con la evaluación de los resultados, es importante señalar que el código que creamos en la imagen 11 no proporcionó una solución. Esto se debe a que la versión del solver Gurobi que utilizamos tiene un límite en el número de variables que puede manejar, y el dataset de Solomon contiene demasiadas variables. Para abordar este problema, reducimos el número de variables y matrices. Sin embargo, al no tener todos los caminos necesarios para calcular las distancias, el solver no pudo encontrar una solución óptima.

Ahora, al analizar las soluciones de los códigos creados por nosotros (imagen 10), que utilizamos la heurística ILS, y el de ChatGPT (imagen 12), que empleó un algoritmo genético, observamos lo siguiente: nuestro código tomó casi 3 segundos en tiempo de ejecución y proporcionó la distancia recorrida. Aunque el código de ChatGPT tuvo un tiempo de ejecución menor, la distancia recorrida resultó ser mayor. Por lo tanto, a pesar de tener un mayor tiempo de ejecución, nuestro código es superior en términos de la calidad de la solución obtenida.

Conclusión:

Estos problemas NP-Hard presentan una serie de características clave, como ventanas de tiempo, capacidades de los vehículos, demanda de los clientes y restricciones de rutas, que requieren soluciones óptimas para minimizar costos y maximizar la eficiencia.

El uso de heurísticas, metaheurísticas y estrategias de optimización en la resolución de estos problemas demuestra la diversidad de enfoques para encontrar soluciones factibles y cercanas al óptimo global. La comparativa entre diferentes códigos y estrategias resalta la importancia de la selección de criterios de optimización y la influencia en los resultados obtenidos.

Además, la formulación matemática proporciona un marco teórico sólido para abordar estos desafíos computacionales, lo que requiere técnicas avanzadas y modelos precisos para encontrar soluciones efectivas.

Y aunque ChatGPT no entregará un resultado optimizado, si no que buscas un resultado que tenga una rapidez destacable, puede ser una buena opción el utilizar esta IA.

Bibliografía:

- Solomon problems
- ChatGPT
- https://repositorio.utp.edu.co/server/api/core/bitstreams/3e902645-43b0-4a32-b898-fcf71d2c52a0/content
- https://github.com/BluSwing/Proyecto-Optimizaci-n
- https://ingenieriaindustrialonline.com/investigacion-de-operaciones/problema-de-enrutamiento-de-vehiculos-capacitados-cvrp-con-google-or-tools/