Universidad San Francisco de Quito

Ciencia de datos

Reporte Proyecto Final

Nombres:

Patricia Lema

Ana Navas

Diana Córdova

Sebastián Ruiz

Materia: Inteligencia Artificial

Fecha: 16 de Mayo del 2025

Contenido

Resumen corto:

Abstract: Este proyecto plantea una serie de soluciones al desafío clásico de optimización combinatoria VRP. El VRP usa aplicaciones en logística y distribución, donde se busca minimizar el costo total de recorrido de una flota de vehículos. Las restricciones a considerar son: capacidad, cobertura de demanda y número de rutas. Este trabajo implementa una estrategia basada en Large Neighborhood Search (LNS) complementada por 2-opt y técnicas de agrupamiento previo, como K-Means, para optimizar rutas en distintas escalas del VRP (small, medium, large). La mejora a la solución inicial se logra mediante iteraciones de destrucción parcial y reparación voraz, permitiendo al algoritmo reconfigurar la asignación de clientes a vehículos y escapar de los óptimos locales.

Los resultados muestran que LNS logra reducir la distancia total recorrida entre un 20% y 50% dependiendo de la configuración, sin aumentar el número de vehículos. Se evidenció que una mala inicialización tiene impacto en el modelo (por ejemplo, con heurísticas greedy puras) y limita la eficacia de métodos locales como 2-opt por sí solos. Para los VPR más extensos, una inicialización guiada por K-Means mejora la convergencia global del sistema, complementado por cross. El análisis cuantitativo fue complementado por visualización de rutas y evaluación de eficiencia en términos de carga utilizada y distancia promedio por vehículo. Se concluye que la combinación de técnicas locales y globales dentro de una arquitectura metaheurística permite abordar el VRP de forma efectiva, demostrando que enfoques de inteligencia artificial orientados a la optimización pueden lograr mejoras sustanciales en problemas reales de planificación logística.

1. Introducción:

El **Problema de Ruteo de Vehículos (VRP)** es un problema clásico de optimización combinatoria en logística y distribución en el que se deben determinar las rutas óptimas para que una flota de vehículos atienda a un conjunto de clientes (con demandas conocidas) partiendo de uno o varios depósitos metrobi.commedium.com. El objetivo típico es minimizar un costo total, por ejemplo la distancia o tiempo total recorrido por todos los vehículos, cumpliendo a la vez ciertas restricciones reales: capacidad limitada de cada vehículo, ventanas de tiempo de entrega, número máximo de vehículos, entre otras. El VRP generaliza el Problema del Viajante (TSP) e incorpora múltiples vehículos y restricciones adicionales como la capacidad, por lo que **es significativamente más complejo (Wenyi, 2023)**.

Desde el punto de vista computacional, el VRP es un problema intratable en el sentido de la complejidad computacional. Formalmente, se ha demostrado que el VRP es un problema de optimización combinatoria NP-hard (no polinomialmente resoluble) (Shi y Niu, 2023). Esto significa que el número de posibles soluciones crece exponencialmente con el número de clientes y vehículos, lo que hace inviable enumerar todas las rutas posibles. En consecuencia, los métodos exactos (com brute force) solo pueden optimizar instancias pequeñas. Para las instancias de tamaño moderado o grande, es necesario recurrir a heurísticas y metaheurísticas que encuentran soluciones muy buenas en tiempo razonable, aunque no garanticen optimalidad global.

2. Enfoque de Solución Utilizado en el Notebook

Small VRP - Medium VRP

- **Solución inicial heurística:** Primero se genera una solución factible inicial de manera heurística (una solución "base"). En el código, se distribuyen los clientes de forma equitativa entre los vehículos disponibles y convergen en el nodo que representa al depósito. A cada ruta inicial se le aplica una optimización local de 2-opt para acortar el circuito de ese vehículo. El resultado es una solución válida donde cada vehículo realiza un recorrido desde el depósito, visita cierto subconjunto de clientes y retorna al depósito. Esta solución inicial **no es óptima** pero sirve de punto de partida.

Aplicación de Metaheurística con Large Neighborhood Search:

Sobre la solución actual, el algoritmo LNS realiza iterativamente los siguientes pasos clave:

1. **Destrucción parcial (*destroy*):** Se selecciona aleatoriamente un subconjunto de clientes (por ejemplo, para el VRP pequeño fue de 30% y para el mediano fue de 20% de los clientes en la implementación) y se **eliminan de sus rutas actuales**. Esto deja huecos en algunas rutas.
2. **Reparación (*repair*):** A continuación, se vuelven a insertar los clientes removidos en la solución. La reinserción se hace de forma **greedy** (voraz), escogiendo para cada cliente removido la ruta y posición que incremente lo menos posible la distancia total. Se respeta la capacidad de cada vehículo al

reubicar clientes, y si algún cliente no cabe en ninguna ruta existente se puede abrir una ruta nueva (usando un vehículo disponible sin asignar).

3. Después de reinsertar todos los clientes, se aplica nuevamente 2-opt en cada ruta para poder pulir la ruta internamente y asegurarnos que no queden optimizaciones pendientes. Este proceso de destruir/reparar equivale a explorar un *gran vecindario* de la solución actual: se prueban nuevas asignaciones de varios clientes simultáneamente, algo que las búsquedas locales simples (como 2-opt por sí solo) no logran porque estas solamente mueven un par de aristas.
4. **Aceptación y memoria:** Si la solución reparada obtenida tiene menor distancia total que la solución actual, entonces se acepta como nueva solución corriente (reemplazando a la anterior). Si además mejora la mejor solución encontrada hasta el momento, se actualiza este óptimo histórico. Si la nueva solución no mejora, puede descartarse o tomarse con cierta probabilidad; en la implementación presentada, se optó por un criterio **greedy estricto** (solo se aceptan movimientos si mejoran la distancia) y se cuenta el número de iteraciones sin. El algoritmo itera este ciclo de destruir&reparar múltiples veces (hasta un máximo predefinido, e.g. 200 iteraciones, o hasta que haya 50 iteraciones sin mejora).

Large VRP:

1. En el caso del problema más grande, se decidió inicializar el número de vehículos con un algoritmo voraz pero esto dio rutas ineficientes. Por lo tanto, se optó por el siguiente método: se realizó una agrupación utilizando K-nearest neighbors a fin de definir grupos de clientes que estén cerca entre sí como parte de una ruta. Posteriormente a esto se aplicó Large Neighborhood Search [1] seguido de crossover exchange con el fin de optimizar globalmente. Una vez hecho esto se utilizaron 2-opt[2] y backward-tracking como métodos para optimizar rutas particulares.
2. El uso de crossover exchange está inspirado en algoritmos genéticos, que intercambian fragmentos de rutas entre diferentes soluciones y permite combinar rutas buenas parciales para una mejor solución. Esto ayuda a introducir variación estructural controlada

3. El uso de 2-opt ayuda a que cada ruta individual esté pulida internamente. El uso de Backward-tracking revisa y revierte decisiones previas (lo que no debe confundirse con aprendizaje) y hace un tipo de "control de calidad" para evitar degradaciones.

3. Métricas de evaluación:

Los **criterios cuantitativos** considerados incluyen la distancia total recorrida por todos los vehículos, el número de vehículos usados y el tiempo de cómputo requerido. A continuación, se resumen las mejoras observadas:

- **Reducción de la distancia total:** La metaheurística LNS logró reducir de forma notable la longitud total de las rutas respecto a la solución inicial. Por ejemplo, para una estancia "pequeña" con 10 clientes y 2 vehículos de capacidad dada, la solución inicial tenía una distancia total de **848.68** (unidades de distancia). Tras la optimización, la distancia total bajó a **685.13**, lo que representa una mejora (reducción) del **19.3%**. En otra configuración con 3 vehículos, la distancia pasó de **1291.13** inicialmente a **946.92** tras LNS (mejora del **26.7%**), y con 4 vehículos de **1350.99** a **921.19** (mejora del **31.8%**) file-mgcyhjc9mcezi1hudjhywbfile-mgcyhjc9mcezi1hudjhywb. Estas cifras nos demuestran una **reducción sustancial del costo** gracias a la optimización. La ruta inicial "greedy" era razonable pero dejó margen considerable para recortar distancia; el algoritmo explotó ese margen reubicando clientes de manera más eficiente entre vehículos y ordenando mejor sus visitas.
- **Uso de vehículos y distribución de carga:** En las instancias de prueba pequeñas, el número de vehículos utilizados en la solución optimizada se mantuvo igual que en la solución base, ya que todos los vehículos disponibles ya estaban siendo utilizados y las demandas requerían usar, por ejemplo, 2 de 2 vehículos (o 3 de 3). Sin embargo, la optimización logró **balancear mejor la carga** entre ellos: en el ejemplo de 2 vehículos, uno de los camiones terminó atendiendo 5 clientes y el otro 4, ambos aprovechando el 90% de su capacidad, mientras que en la solución inicial la distribución podría haber sido menos equilibrada.
- **Tiempo de cómputo:** La versión optimizada logró estas mejoras con **tiempos de cómputo muy bajos**, gracias a la eficiencia de las heurísticas. En las instancias pequeñas, el algoritmo LNS converge en fracciones de segundo. Por ejemplo, para 2 vehículos el proceso completo tardó apenas **0.03 segundos** en encontrar la mejor

opción. Incluso en problemas más grandes (docenas de clientes), los tiempos reportados rondan del orden de 0.5 a 2 segundos. Esto contrasta fuertemente con métodos exactos, que podrían tardar minutos o horas o no escalar a esos tamaños. La rápida convergencia se debe a que cada iteración de LNS realiza inserciones voraces (de costo computacional moderado) y las limita a un número fijo de iteraciones. En la práctica, el algoritmo suele encontrar mejoras significativas en las primeras decenas de iteraciones; una vez estabilizado, se detiene al no hallar nuevas mejoras dentro del límite fijado.

- En conjunto, **la versión optimizada supera ampliamente a la versión base** en cuanto a calidad de la solución, reduciendo la distancia total recorrida sin incrementar los recursos utilizados. La mejora porcentual depende de la instancia, pero en los experimentos oscila entre ~20% y ~30% para VRPs pequeños (y aún mayor en algunos casos de prueba más grandes, llegando a ~50% de reducción de distancia). Esto se logró con un costo computacional bajo, haciendo viable aplicar esta optimización en escenarios reales donde se requiere recalcular rutas rápidamente.

4. Dificultades

- El número de posibles rutas crece exponencialmente con el número de clientes y el tamaño de la flota. Esto hace que el costo computacional para encontrar la solución óptima sea demasiado alto.
- Si la inicialización de las rutas no se realiza con una heurística o algoritmo en mente, se vuelve difícil que métodos de optimización aplicados posteriormente puedan llegar a un espacio óptimo de soluciones, incrementando el costo computacional.
- La restricción de la capacidad del vehículo de entrega hace que sea necesario verificar durante el proceso de optimización que la demanda de los clientes a lo largo de la ruta no exceda la capacidad del vehículo.
- Ciertos casos no eran solvibles debido a que la demanda total de los clientes excedía la capacidad total de la flota.
- El archivo para el subcaso de 4 vehículos dentro del caso small tenía un error de sintaxis que debe ser corregido antes de leer el archivo.
- Existe el riesgo de que el proceso se atore en un mínimo local, por lo que se debe implementar métodos que puedan empujar al algoritmo fuera de esa región para que pueda explorar otra región del espacio de soluciones.

- Parte del proceso de determinar la eficiencia de las rutas consiste en inspeccionar visualmente el gráfico para analizar el resultado y ver donde puede haber mejoras.
- A pesar de que usar el algoritmo de agrupamiento por cercanía para iniciar rutas que recorran clientes que estén cerca entre sí, fue necesario asegurarse de que las rutas generadas no excedan los límites de capacidad de los vehículos.

5. Conclusiones

- EL esquema LNS es una **metaheurística** que explora aleatoriamente un gran espacio de soluciones (distintas distribuciones de clientes en vehículos) lo que es ideal para escapar de óptimos locales pequeños.
- El LNS permite que un cliente cambie de ruta de un vehículo a otro durante la fase de búsqueda global, gracias a la destrucción parcial de la solución. Esto permite reubicar clientes entre distintas rutas, logrando una optimización más global.
- La solución del notebook emplea *heurísticas* (2-opt, inserción voraz) para las decisiones locales, envueltas en una *metaheurística LNS* que guía la búsqueda global rompiendo y reconstruyendo parcialmente la solución.
- No se utiliza ningún algoritmo genético, colonia de hormigas u otro método poblacional; el enfoque es más cercano a un búsqueda local iterada especializada para VRP. Tampoco se aplica un método exacto clásico (como programación lineal entera), debido a la complejidad ya mencionada del VRP para tamaños mayores.
- En el VRP más grande, donde inicialmente quizás no todos los vehículos estaban completamente cargados, el algoritmo LNS conseguía que algunos vehículos cubran todas las entregas, dejando **rutas vacías** y vehículos sobrantes. Por ejemplo, se permitían hasta 8 vehículos pero la demanda podía satisfacerse con 7, el algoritmo efectivamente dejó 1 vehículo sin ruta, usando solo 7 y reduciendo así la flota. Esto significa beneficios para la empresa en la práctica, pues implica menos vehículos en la calle y menor costo operativo.
- Usamos K-Means como un método de clustering inicial para **agrupar espacialmente los clientes** y así crear rutas base más coherentes. Esto mejora la calidad de las soluciones iniciales, reduce el trabajo de los algoritmos de optimización y facilita que métodos como LNS o 2-opt converjan hacia soluciones más eficientes

Finalmente, este ejercicio de tipo híbrido mostró que mediante inteligencia artificial (en forma de heurísticas avanzadas) se pueden abordar problemas de ruteo complejos de manera efectiva.

La solución basada en búsqueda y optimización logró rutas más cortas (ahorro de costos y tiempo) sin hardware extra ni soluciones externas, sólo mediante mejor cálculo.

Para futuras mejoras se podría incorporar otras metaheurísticas (p. ej., búsquedas tabú, algoritmos genéticos). Esto significa adoptar un enfoque poblacional y evolutivo: se perdería el control directo sobre la estructura inicial (como con K-Means), pero se ganaría diversidad, capacidad de escape de óptimos locales y una forma poderosa de combinar soluciones parciales buenas.

No obstante, lo implementado ya enseña un mensaje poderoso: combinar técnicas de búsqueda local con estrategias metaheurísticas globales produce soluciones de alta calidad en VRP

6. Referencias:

Shi, Ruiyang, and Lingfeng Niu. 2023. "A Brief Survey on Learning Based Methods for Vehicle Routing Problems." *Procedia Computer Science* 221:773–80.

https://doi.org/10.1016/j.procs.2023.08.050.

Wenyi, Lu. 2023. "Research on Vehicle Routing Problem and Application Scenarios." In *Handbook of Mobility Data Mining*, 63–88. Elsevier.

https://doi.org/10.1016/B978-0-323-95892-9.00006-1.

Distribución del trabajo:

Script de VRP	Sebastián Ruiz
Script de Agente RAG	Diana Córdova
Script de Packing Problem	Patricia Lema
Presentación - elaboración	Diana Córdova, Sebastián Ruiz, Ana Navas.
Presentación - oral	Ana Navas, Patricia Lema, Sebastián Ruiz, Diana Córdova
Reporte final	Diana Córdova, Sebastián Ruiz