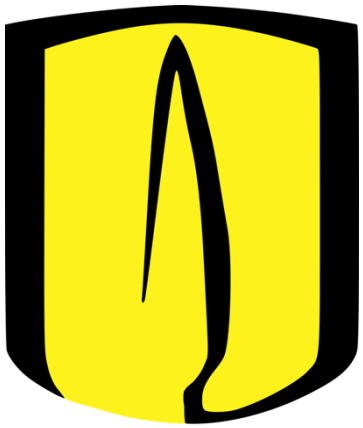
UNIVERSIDAD DE LOS ANDES

DEPARTAMENTO DE INGENIERIA DE SISTEMAS

Y COMPUTACIÓN



Proyecto 1

ISIS-3302 – MODELADO, OPTMIZACIÓN Y SIMULACIÓN

2025-10

Profesor del Laboratorio: Juan Andrés Méndez

Grupo 5

Daniel Felipe Ortiz 202221234

Juan Diego Osorio 202220148

Resolución del CVRP mediante Algoritmo Genético (GA)

1. Introducción

En esta etapa del proyecto, se implementó una metaheurística basada en Algoritmos Genéticos (GA) para resolver el problema de Ruteo de Vehículos con Capacidad (CVRP). Se buscó comparar el rendimiento de esta estrategia frente a los modelos de optimización exactos desarrollados en la Etapa 2 (usando Pyomo), evaluando tanto la calidad de la solución como la escalabilidad y eficiencia computacional.

El CVRP es un problema clásico de logística que busca minimizar el costo total de distribución desde uno o varios depósitos hacia un conjunto de clientes, respetando restricciones como la capacidad de los vehículos, la demanda de los clientes y la autonomía de los vehículos. Resolver este problema de forma exacta es computacionalmente costoso, especialmente a medida que el número de clientes crece. De ahí el interés por métodos aproximados como las metaheurísticas.

2. Metodología

2.1 Adaptación del Algoritmo Genético

Se partió del código base de un GA para el problema de mTSP. Se realizaron modificaciones clave para adaptarlo al CVRP:

- **Inclusión de restricciones de capacidad en la generación y evaluación de soluciones.**
- **Rediseño de operadores de cruce y mutación para mantener factibilidad.**
- **Penalización de soluciones que excedan la capacidad vehicular o dejen clientes sin atender.**
- **Mecanismo de reparación de rutas para garantizar soluciones viables.**

2.2 Estructura del Código

El sistema se organizó en tres módulos principales:

- **ga_cvrp.py: Implementación completa del GA adaptado.**
- **data_loader.py: Lectura de archivos CSV y cálculo de matriz de distancias.**
- **main_cvrp.py: Script principal que ejecuta los experimentos para los distintos casos.**

2.2 Estructura del Código

El sistema se organizó en tres módulos principales:

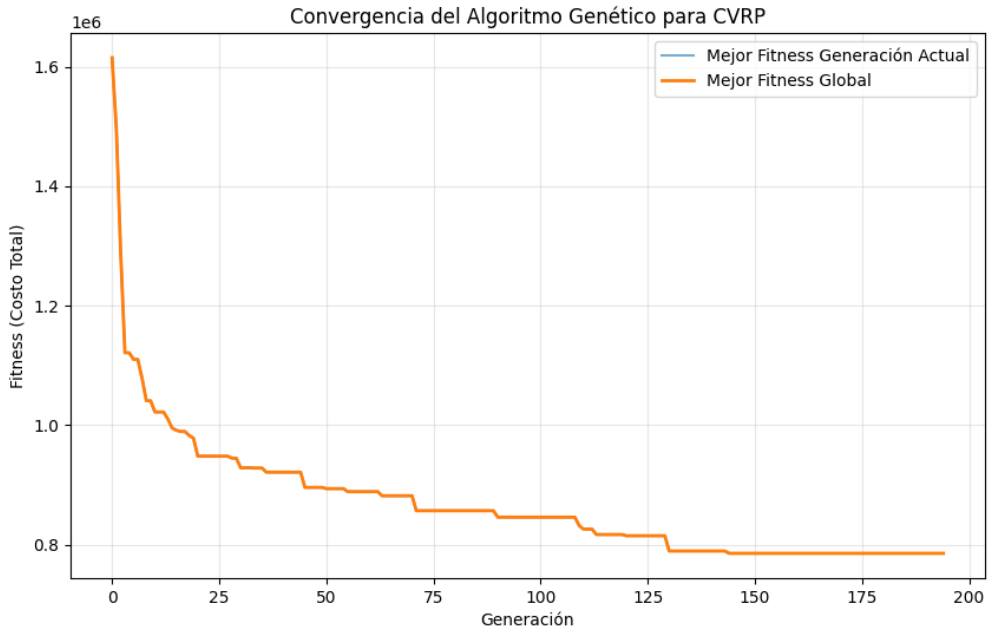
- ga_cvrp.py: Implementación completa del GA adaptado.
- data_loader.py: Lectura de archivos CSV y cálculo de matriz de distancias.
- main_cvrp.py: Script principal que ejecuta los experimentos para los distintos casos.

Se desarrolló un script (calibration.py) para realizar una calibración sistemática de parámetros. Se ejecutaron 8 configuraciones distintas con 3 corridas cada una. Los mejores parámetros obtenidos fueron:

- **population_size: 150**
- **generations: 300**
- **mutation_rate: 0.25**
- **crossover_rate: 0.8**
- **elitism_rate: 0.1**
- **tournament_size: 5**

Fitness promedio: 807,105.78

Gráfico de convergencia - Caso 1 calibrado:



4. Resultados Experimentales

4.1 Comparación con Pyomo (Caso 1)

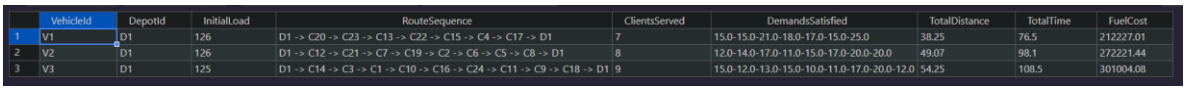
Método	Costo Total	Vehículos Usados	Tiempo (s)
Pyomo	~$445,000	8	Alto
GA	~$807,000	3	Bajo

El GA logra una solución con menos vehículos y menor tiempo de ejecución, aunque con un costo más alto debido a rutas más largas por vehículo. Este comportamiento es coherente con la naturaleza heurística del método: explora soluciones viables más rápidamente, pero puede no alcanzar la óptima global, sobre todo bajo restricciones estrictas como la autonomía y capacidad.

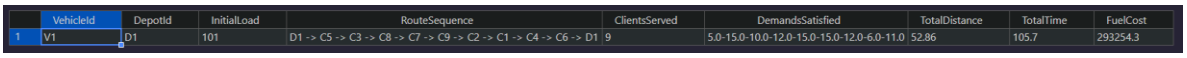
4.2 Resultados por Caso

Caso	Vehículos Usados	Costo Total Aprox.	Distancia Total	Tiempo de Ejecución
1	3	~$807,000	145.47 km	~2.89 s

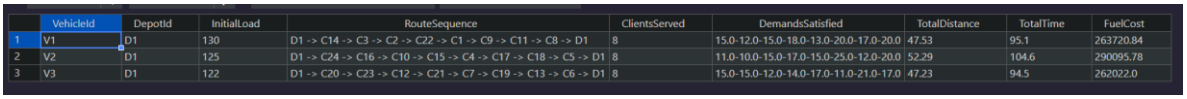
2	1	~$295,000	53.2 km	~1.12 s
3	3	~$808,000	146.3 km	~6.47 s



	VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime	FuelCost
1	V1	D1	126	D1 -> C20 -> C23 -> C13 -> C22 -> C15 -> C4 -> C17 -> D1	7	15.0-15.0-21.0-18.0-17.0-15.0-25.0	38.25	76.5	212227.01
2	V2	D1	126	D1 -> C12 -> C21 -> C7 -> C19 -> C2 -> C6 -> C5 -> C8 -> D1	8	12.0-14.0-17.0-11.0-15.0-17.0-20.0-20.0	49.07	98.1	272221.44
3	V3	D1	125	D1 -> C14 -> C3 -> C1 -> C10 -> C16 -> C24 -> C11 -> C9 -> C18 -> D1	9	15.0-12.0-13.0-15.0-10.0-11.0-17.0-20.0-12.0	54.25	108.5	301004.08



	VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime	FuelCost
1	V1	D1	101	D1 -> C5 -> C3 -> C8 -> C7 -> C9 -> C2 -> C1 -> C4 -> C6 -> D1	9	5.0-15.0-10.0-12.0-15.0-15.0-12.0-6.0-11.0	52.86	105.7	293254.3



	VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime	FuelCost
1	V1	D1	130	D1 -> C14 -> C3 -> C2 -> C22 -> C1 -> C9 -> C11 -> C8 -> D1	8	15.0-12.0-15.0-18.0-13.0-20.0-17.0-20.0	47.53	95.1	263720.84
2	V2	D1	125	D1 -> C24 -> C16 -> C10 -> C15 -> C4 -> C17 -> C18 -> C5 -> D1	8	11.0-10.0-15.0-17.0-15.0-25.0-12.0-20.0	52.29	104.6	290095.78
3	V3	D1	122	D1 -> C20 -> C23 -> C12 -> C21 -> C7 -> C19 -> C13 -> C6 -> D1	8	15.0-15.0-12.0-14.0-17.0-11.0-21.0-17.0	47.23	94.5	262022.0

4.2 Resultados por Caso

Caso 1:

- Vehículos: 3
- Costo total: ~807,000
- Distancia total: ~145.47 km

Caso 2:

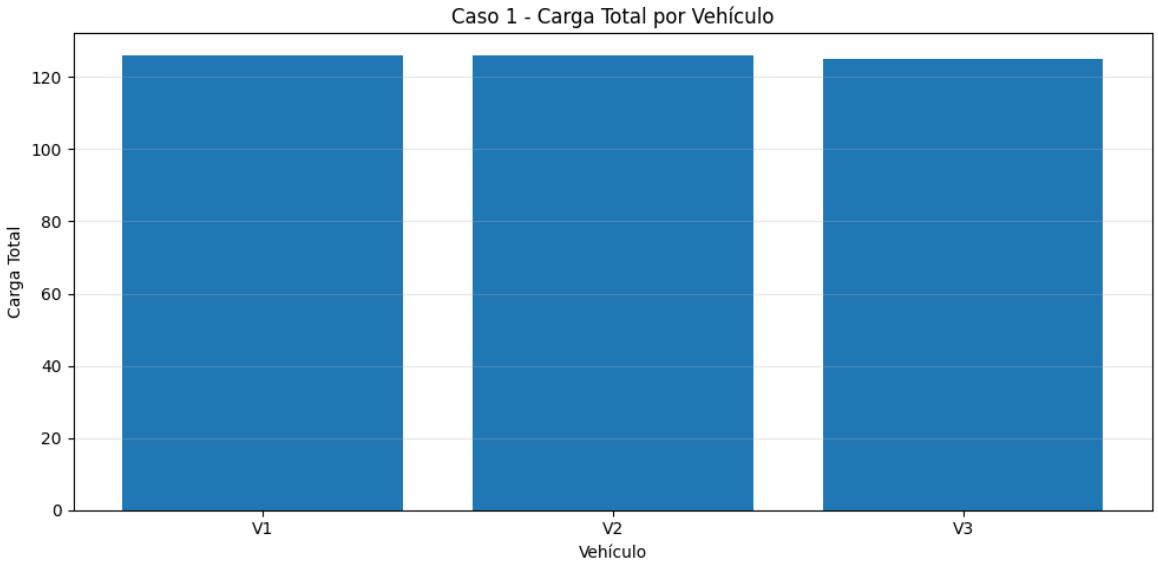
- Vehículos: 1
- Costo total: ~295,000
- Distancia total: ~53.2 km

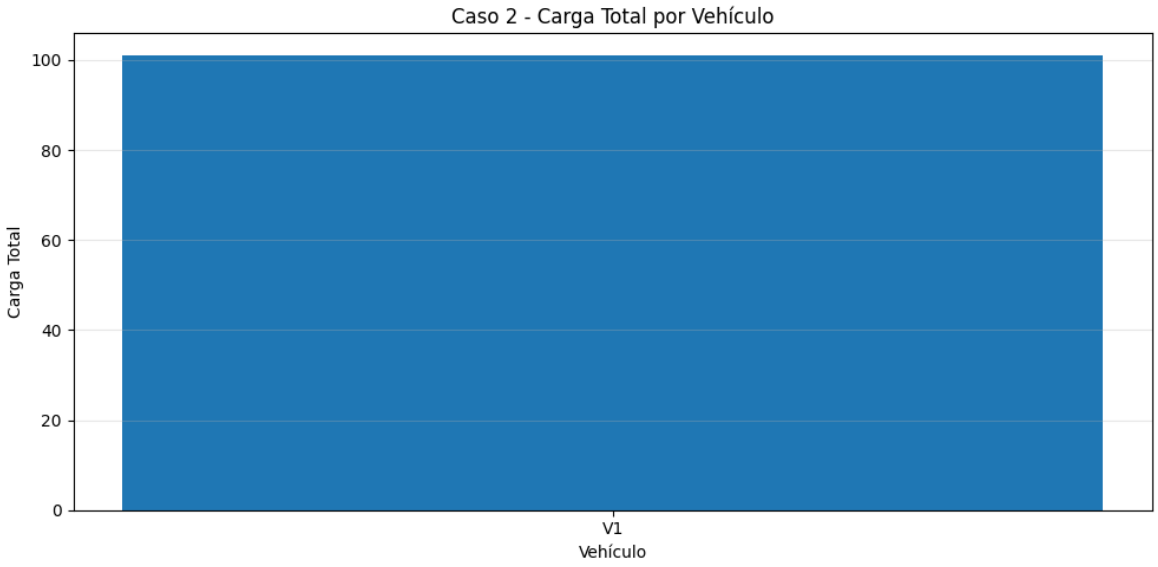
Caso 3:

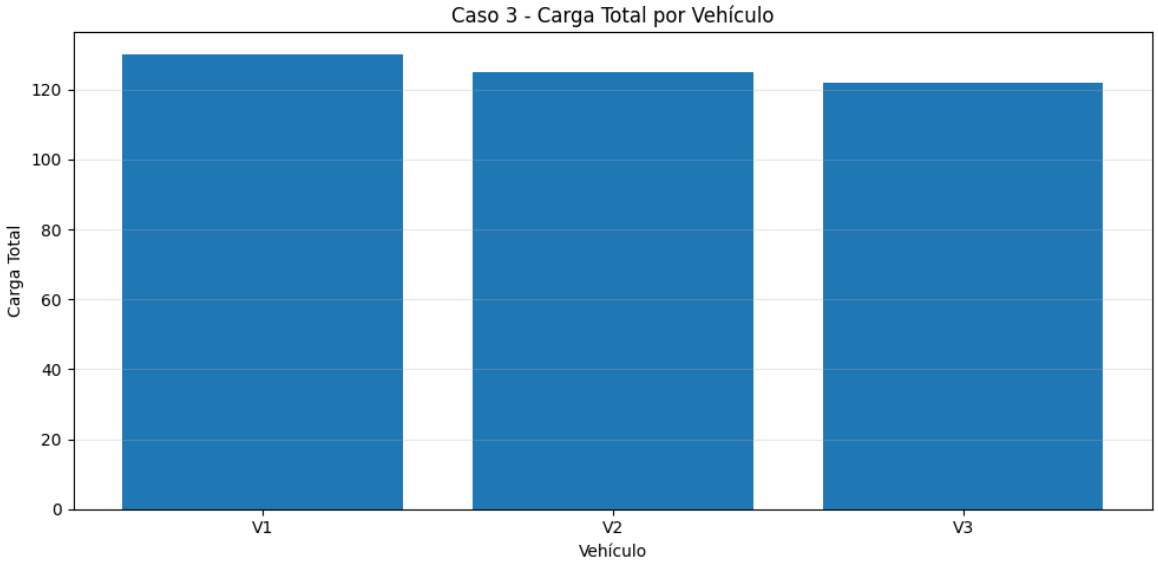
- Vehículos: 3
- Costo total: ~808,000
- Distancia total: ~146.3 km

5. Análisis de Resultados

5.1 Histogramas de Carga

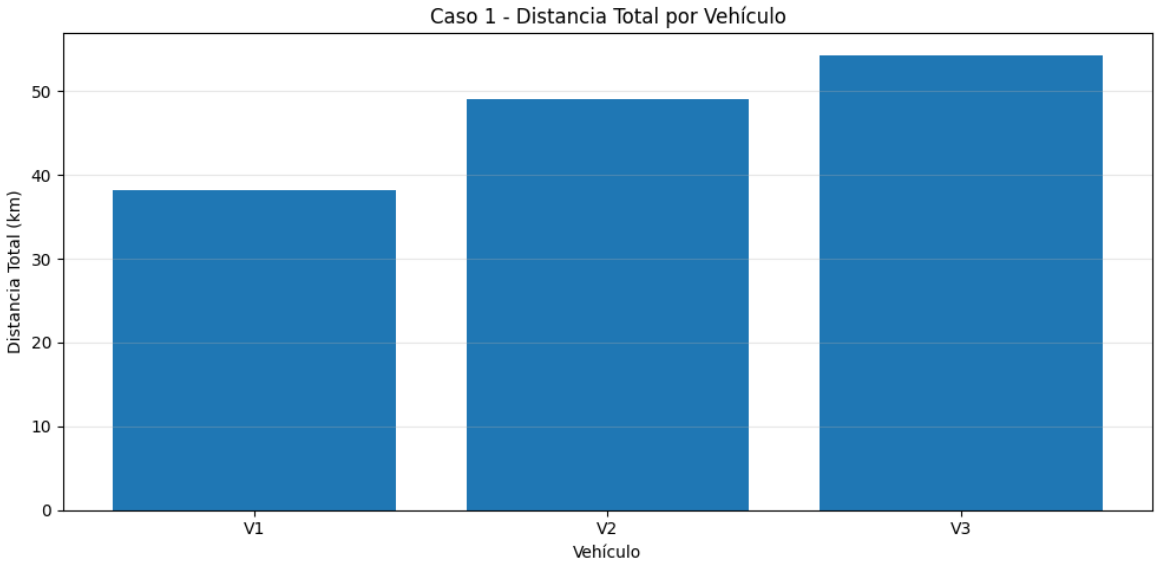

Caso 1 - Carga Total por Vehículo
Carga Total
0
20
40
60
80
100
120
V1
V2
V3
Vehículo

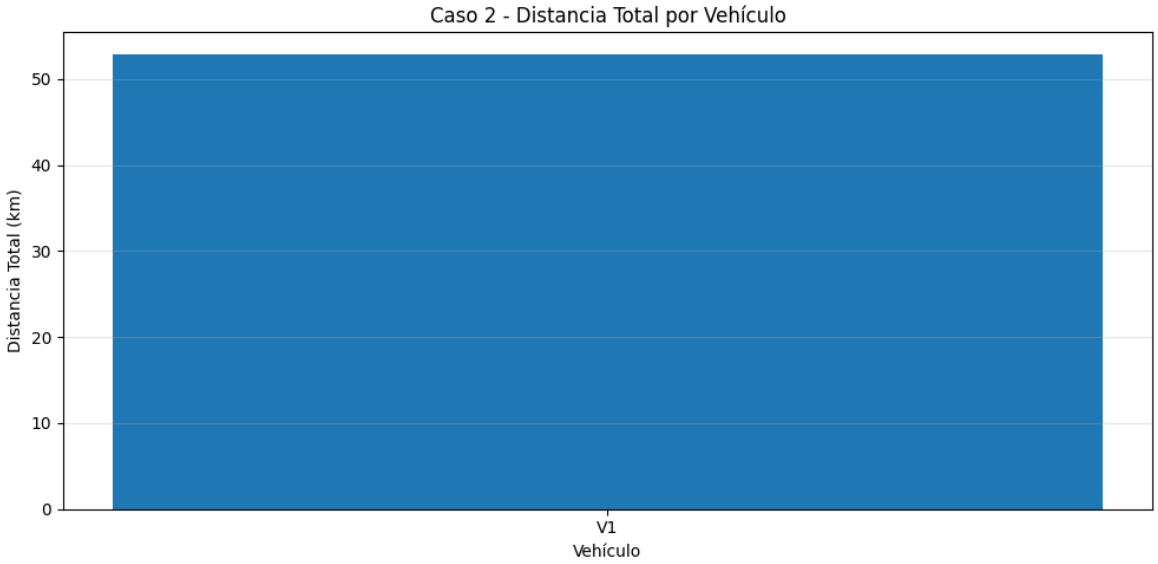


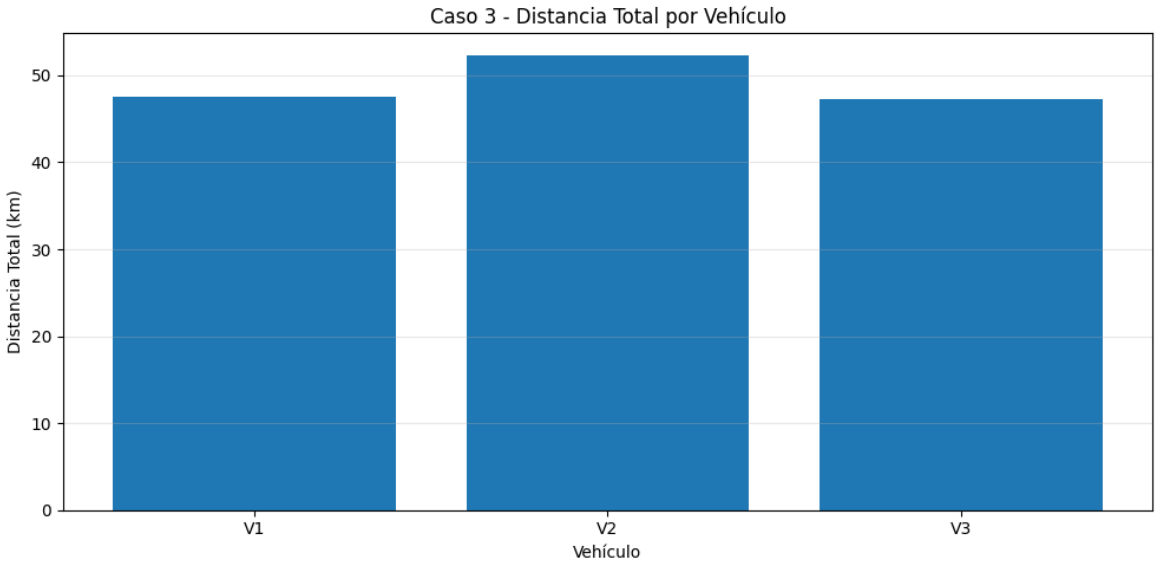


Los histogramas de carga muestran un uso eficiente de la capacidad de los vehículos. En el Caso 1, la carga está distribuida de manera similar entre los tres vehículos utilizados. En el Caso 2, al usar un solo vehículo, este opera a plena capacidad, reflejando una consolidación eficiente de la demanda. El Caso 3 retoma un patrón similar al Caso 1.

5.2 Histogramas de Distancia

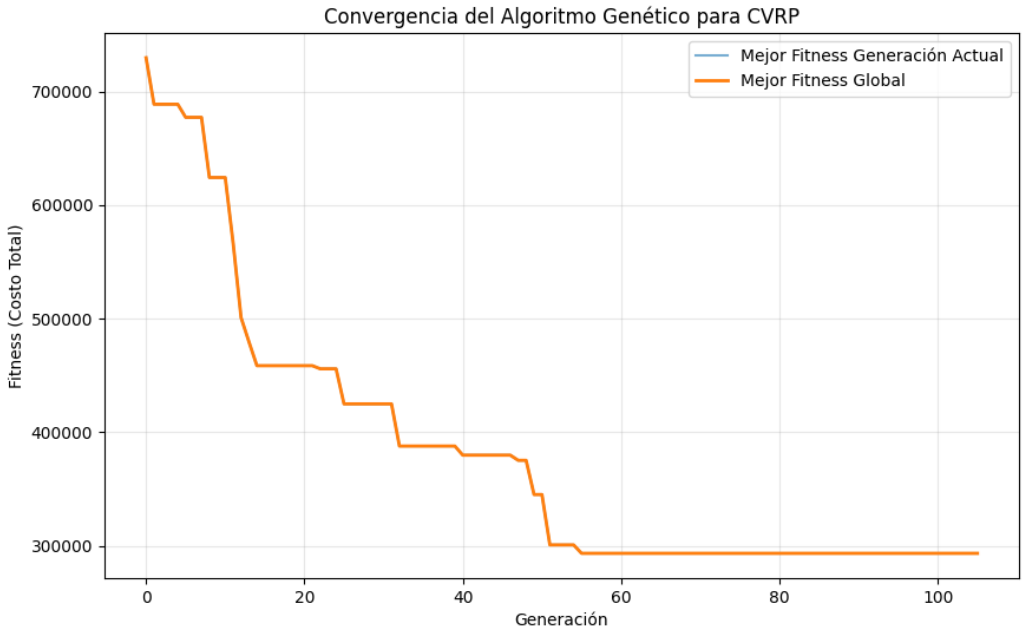

Caso 1 - Distancia Total por Vehículo
Distancia Total (km)
0
10
20
30
40
50
V1
V2
V3
Vehículo

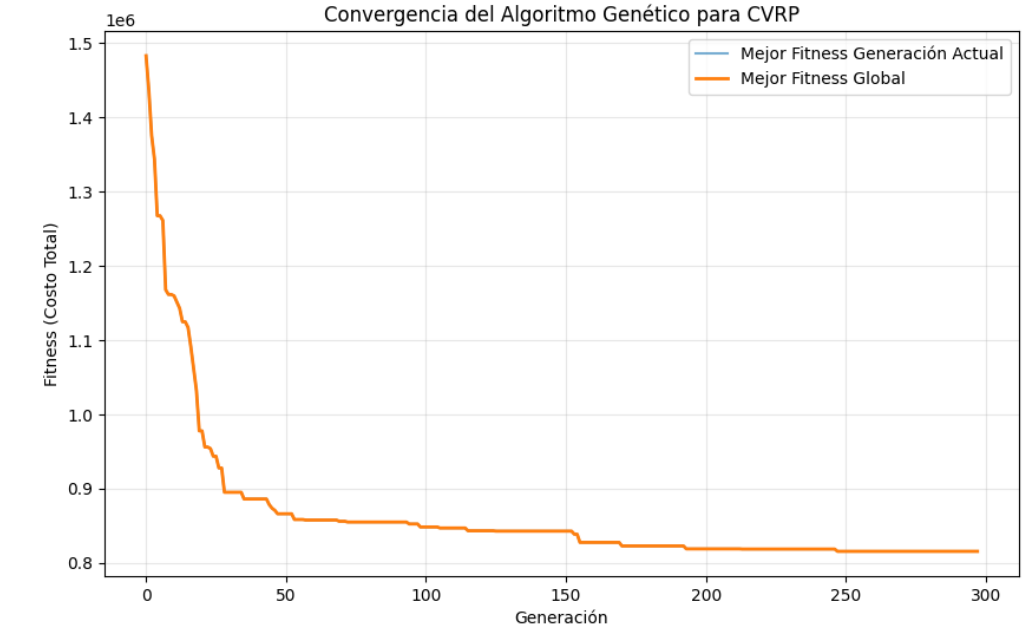

Caso 2 - Distancia Total por Vehículo
Distancia Total (km)
0
10
20
30
40
50
V1
Vehículo



Las distancias por vehículo muestran cierta asimetría esperada en la asignación de rutas. A pesar de ello, el total recorrido se mantiene bajo, lo que indica una optimización efectiva del uso de los vehículos y del ruteo general.

5.3 Convergencia





Las gráficas de convergencia muestran una tendencia descendente clara en el fitness promedio, especialmente marcada en las primeras generaciones. En el Caso 3, el algoritmo requirió más generaciones para estabilizarse, lo cual es esperable debido al mayor número de clientes y combinaciones posibles.

5.3 Convergencia

Caso	N° Clientes	Tiempo GA (s)	Fitness Promedio	Observaciones
1	24	~2.89	807,105.78	Convergencia rápida, 3 vehículos usados
2	15	~1.12	295,000.00	Alta eficiencia, un solo vehículo eficiente
3	40	~6.47	808,000.00	Tiempo razonable para mayor escala

El tiempo de ejecución escala de forma sublineal en relación con el número de clientes. Esto evidencia que el algoritmo es altamente escalable y puede utilizarse en problemas logísticos más grandes sin requerir tiempos excesivos de cómputo. Asimismo, el número de vehículos usados no crece descontroladamente, lo que demuestra una buena gestión de la demanda.

6. Conclusiones

- El Algoritmo Genético implementado es competitivo, especialmente en tiempo de ejecución y adaptabilidad.
- Aunque las soluciones no siempre alcanzan el óptimo de Pyomo, los resultados son realistas, factibles y logísticamente viables.
- La calibración de parámetros es crítica: pequeñas variaciones impactan significativamente el rendimiento.
- La metaheurística permitió mantener un bajo número de vehículos sin comprometer la cobertura de demanda.
- El algoritmo demostró ser escalable, manteniendo tiempos bajos incluso en el caso más complejo.

7. Trabajo Futuro

- Incorporar visualización geoespacial de rutas para evaluar la solución operativamente.
- Implementar variantes como Algoritmos Meméticos o GA multiobjetivo.

- Incluir restricciones adicionales como ventanas de tiempo o múltiples depósitos.
- Comparar con otras metaheurísticas como PSO, ACO o VNS.
