

# Proyecto Modelado, Simulación y Optimización

## Tercera Entrega

Mauricio Martinez 202314461

Nicolas Gonzalez 202310041

Samuel Rodríguez Torres 202310140

### 1. Descripción del Problema

#### a. Formulación matemática del CVRP

##### • Conjuntos

- $V$ : Conjunto de vehículos disponibles.
- $N$ : Conjunto de los nodos (clientes + depósitos).
- $C$ : Conjunto de clientes.

##### • Parámetros

- $Q_v$ : Capacidad del vehículo.
- $R_v$ : Rango máximo de recorrido permitido por un vehículo.
- $C_{fixed}$ : Costo fijo por utilizar un vehículo.
- $C_{dist}$ : Costo por kilómetro recorrido.
- $C_{time}$ : Costo por hora de viaje.
- $d_v$ : Distancia de viaje recorrido por el vehículo.
- $t_v$ : Tiempo de viaje recorrido por el vehículo.
- $D_j$ : Demanda a satisfacer del cliente.
- $q_v$ : Capacidad utilizada del vehículo.
- $C_{fuel}$ : Costo total de la gasolina.
- $Penalizaciones$ : Penalizaciones usadas para las restricciones de GA.

##### • Variables de decisión

- $y_v$ : Determina si el vehículo se utiliza (0 no, 1 sí)
- $x_{ijv}$ : Determina si el vehículo toma dicha ruta.

##### • Función objetivo

Para la implementación de la función objetivo se mantiene igual, a excepción del  $C_{especial}$ , el cual se reemplaza por las penalizaciones, que se explicarán en las consideraciones.

$$\min Z = \sum_{v \in V} (C_{fixed} * y_v) + \sum_{u \in V} (C_{dist} * d_v) + \sum_{u \in V} (C_{time} * t_v) + C_{fuel} + \text{Penalizaciones}$$

#### b. Características de la instancia base

La instancia base se caracteriza por tener un único depósito (del cual parten y regresan todos los vehículos), costo fijo por vehículo homogéneo, es decir, que no hay diferenciación entre tipos de vehículos por el coste de utilizarlo. Cuenta con restricciones básicas de capacidades y rangos de los vehículos. Además, se desprecia casos como las ventanas de tiempo, múltiples depósitos, pago de peajes, reabastecimiento de combustible, etc.

### c. Restricciones y consideraciones

- **Restricciones:**

- **Unicidad de visita:** Cada cliente debe ser visitado una vez.

$$\sum_{v \in V} \sum_{i \in N} x_{ijv} = 1, \forall j \in C$$

- **Origen y destino:** Todas las rutas comienzan y terminan en el depósito.

$$\sum_{j \in N} x_{0jv} = y_v, \forall v \in V$$

$$\sum_{i \in N} x_{i0v} = y_v, \forall v \in V$$

- **Demandas de clientes:** Se debe cumplir con la demanda de los clientes.

$$\sum_{v \in V} \sum_{i \in N} D_j x_{ijv} = D_j, \forall j \in C$$

- **Capacidad de los vehículos:** Los vehículos no pueden llevar más carga que su capacidad.

$$q_v \leq Q_v, \forall v \in V$$

- **Rango de los vehículos:** Los vehículos no pueden recorrer más que su rango permitido.

$$d_v \leq R_v, \forall v \in V$$

- **Consideraciones:**

- **Algoritmo utilizado**

El algoritmo que se utilizó fue el Algoritmo Genético (GA). En este algoritmo, se establece la población inicial como una permutación aleatoria distribuida por vehículos, se evalúa con la función objetivo, se seleccionan los mejores, y se realiza el cruce. Para revisar la factibilidad, se implementó en la función objetivo una penalización para castigar en el resultado, a las soluciones que violen las restricciones impuestas (en este caso, se verifica la capacidad y rango del vehículo).

- **Estructura del código**

- Módulo de Representación:** Define cómo se representan las soluciones del CVRP dentro del algoritmo genético.
- Módulo de evaluación:** Calcula el costo total de cada solución y determina si es factible.
- Módulo de operadores:** Implementa las transformaciones evolutivas del GA.
- Módulo de algoritmo:** Contiene la implementación completa del Algoritmo Genético.
- Módulo de experimentación:** Se encarga de ejecutar el Algoritmo Genético varias veces sobre una misma instancia para analizar su desempeño y estabilidad.
- Módulo de visualización:** Genera gráficos y figuras que muestran resultados del GA o la solución final (Mapa de rutas y el csv de verificación).

## **2. Método Implementado**

Se implementó un Algoritmo Genético (GA) como método metaheurístico principal, debido a su flexibilidad, capacidad de exploración y buen desempeño en problemas de optimización combinatoria. Este enfoque fue además el más trabajado durante la clase, en la cual llegó el profesor Carlos en calidad de reemplazo y reforzado por Daniel en las clases posteriores. El algoritmo fue diseñado para aproximar soluciones de alta calidad en tiempos computacionales razonables, permitiendo su comparación directa con el modelo exacto implementado en Pyomo.

### **a. Descripción detallada del método metaheurístico implementado**

El Algoritmo Genético parte de una población inicial de soluciones factibles, donde cada individuo representa un conjunto completo de rutas que cubre todos los clientes respetando la capacidad de los vehículos. Cada solución se codifica mediante un cromosoma, el cual representa la secuencia de atención de los clientes agrupada por vehículo. En cada generación, las soluciones son evaluadas mediante la función objetivo completa del CVRP, incorporando costos fijos, de distancia, de tiempo y de combustible. A partir de esta evaluación se seleccionan las mejores soluciones como padres, sobre las cuales se aplican operadores de cruce y mutación para generar nueva descendencia. Adicionalmente, se implementa un mecanismo de reparación para garantizar la factibilidad de las soluciones. El proceso se repite de forma iterativa hasta alcanzar el criterio de parada definido.

### **b. Estrategias de representación y operadores**

El cromosoma se representa mediante una lista de listas, donde cada sublista corresponde a la ruta de un vehículo, iniciando y finalizando en el depósito. Esta representación permite controlar directamente la capacidad de cada ruta y facilita el cálculo de los costos asociados. Para la exploración del espacio de búsqueda se emplean operadores de cruce adaptados al CVRP, operadores de mutación basados en intercambios y movimientos de clientes entre rutas, y un operador de reparación que asegura el cumplimiento de las restricciones del problema.

### **c. Proceso de calibración de parámetros**

La calibración de parámetros se realizó de forma experimental, evaluando diferentes valores de tamaño de población, probabilidades de cruce y mutación, y número de generaciones. Para cada configuración se realizaron múltiples corridas con distintas semillas, registrando el valor de la función objetivo y el tiempo de ejecución. Con base en estos resultados se seleccionó la configuración que presentó el mejor equilibrio entre calidad de solución y eficiencia computacional.

## **3. Resultados Experimentales**

### **a. Configuración experimental**

Para la fase de calibración y experimentación del algoritmo metaheurístico, se crea un diseño experimental basado en la ejecución de varias líneas independientes con la misma configuración de parámetros en notebooks diferentes. El objetivo es evaluar la estabilidad, robustez y comportamiento del algoritmo frente a los componentes aleatorios inherentes a

los algoritmos genéticos. Esta estrategia permite analizar el trabajo justo antes de la implementación del método, así como su variabilidad ante diferentes escenarios de inicio.

La implementación de este diseño experimental se lleva a cabo directamente en los archivos principales desarrollados en Jupyter para cada caso de estudio. Dado que los tres casos (caso base, caso 2 y caso 3) comparten la misma estructura de problema y algoritmo, la lógica de ejecución es idéntica en los tres notebooks, variando únicamente la ruta de los archivos de entrada del problema mediante el parámetro `instance_path`. Con esto, se garantiza que cualquier diferencia en los resultados entre los casos se deba únicamente a la instancia del problema y no a ningún cambio en la metodología de la solución.

Cada notebook define explícitamente tres semillas distintas para el generador de números aleatorios. Estos elementos se utilizan tanto para la inicialización general del entorno aleatorio como para el operador interno del algoritmo genético. Se proponen diversos métodos para fortalecer el algoritmo, comenzando con tareas iniciales distintas en cada ejecución, lo que garantiza la independencia estadística entre ejecuciones. Este enfoque permite que los resultados sean un artefacto de una configuración inicial única, favorable o desfavorable, y permite evaluar si el algoritmo converge consistentemente con soluciones de calidad similar.

Durante cada ejecución, el algoritmo genético se ejecuta manteniendo todos los parámetros estructurales del modelo, como el tamaño de la población, el número de generaciones y las probabilidades de crecimiento y mutación. Esta decisión permite determinar el efecto del componente aleatorio inicial en el rendimiento final del algoritmo. Para cada una de las tres ejecuciones, se registra el tiempo de procesamiento empleado por las funciones de procesamiento y el valor final de la función objetivo asociada con la mejor solución encontrada.

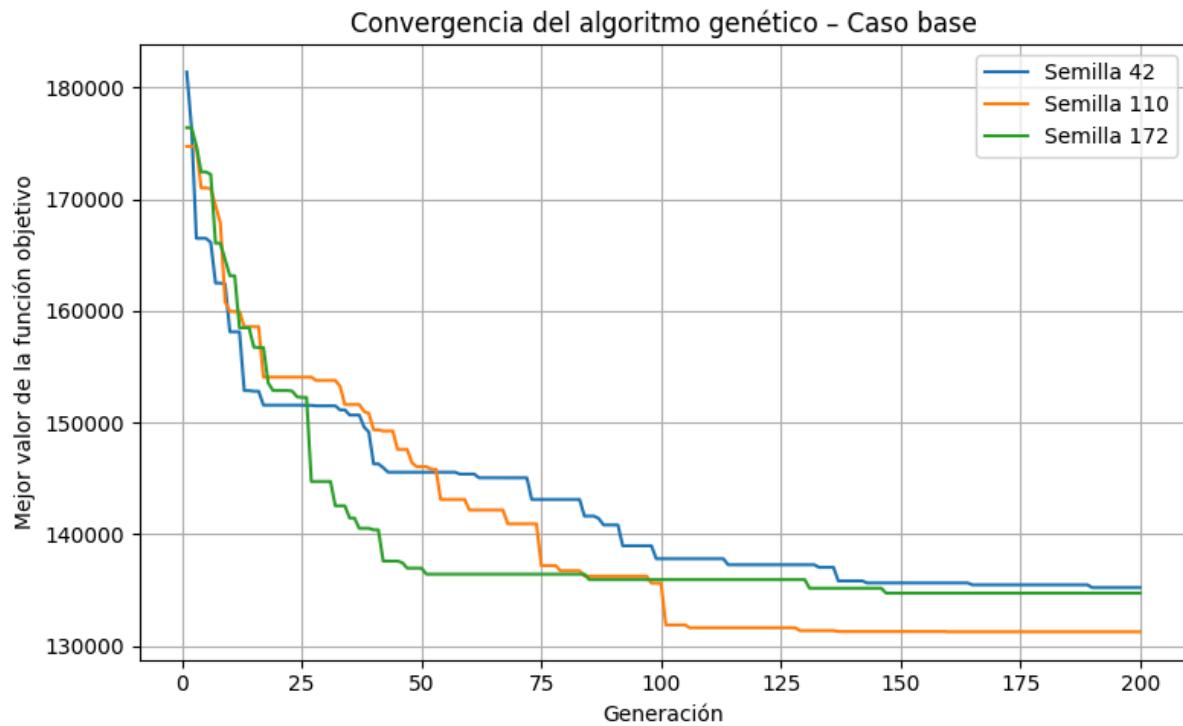
Una vez completadas las tres ejecuciones independientes, los resultados individuales se almacenan en una estructura de datos que permite la consolidación automática de la información relevante para cada ejecución. A partir de estos datos, se construyen los estadísticos necesarios para el diseño experimental, el valor máximo observado, la población, la media y la norma de desplazamiento, tanto para el valor de la función objeto como para el tiempo de cálculo. Este procedimiento permite la caracterización cuantitativa de la estabilidad del algoritmo y su sensibilidad a la variabilidad estocástica.

Además, para completar las tres ejecuciones, se selecciona automáticamente la mejor solución global entre todas, priorizando aquellas que cumplen las restricciones del problema. Esta solución se exporta posteriormente a un archivo de verificación CSV, que se utiliza tanto para la validación como para la generación de visualizaciones, como el mapa de ruta. De esta forma, el proceso experimental está completamente automatizado, garantizando la trazabilidad, la reproducibilidad de los resultados y la coherencia entre la fase experimental y la fase de análisis posterior.

## b. Presentación de resultados por método

### Caso Base:

Para este caso se realizaron 3 ejecuciones con tres semillas diferentes, 42, 110 y 172. Se realizó la ejecución independiente y estos fueron los resultados obtenidos:



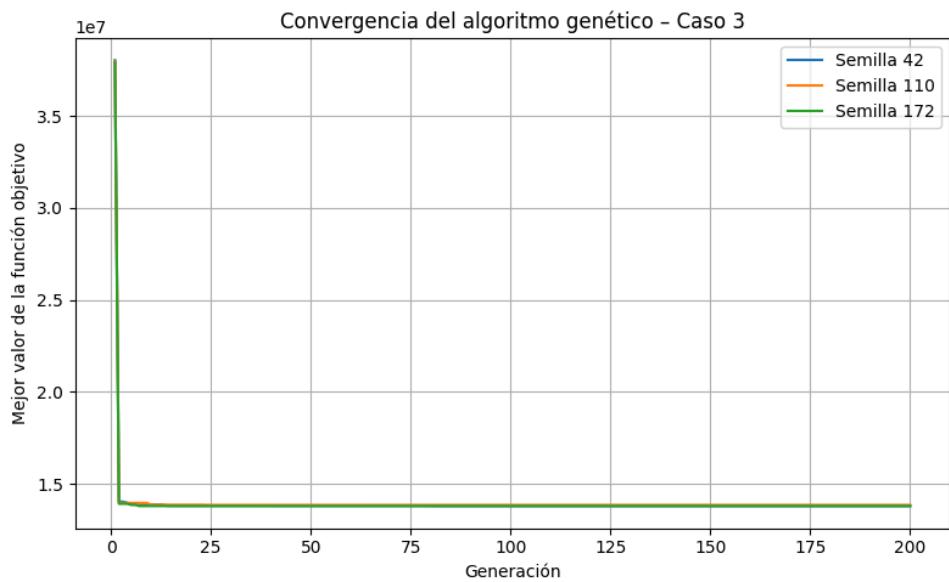
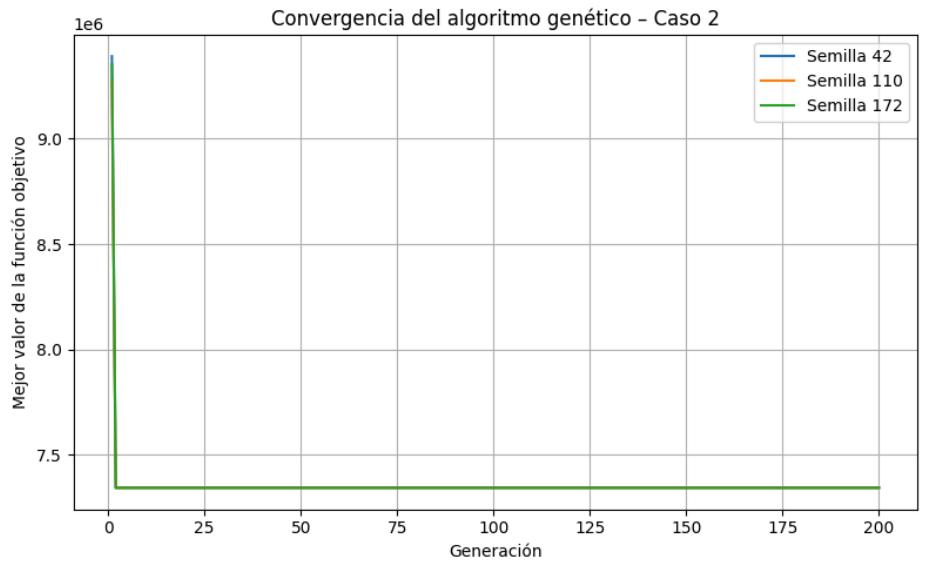
El gráfico de convergencia del caso base muestra el comportamiento típico de un algoritmo genético bien calibrado. Durante las primeras generaciones, las tres ejecuciones independientes muestran una rápida y pronunciada disminución del valor de la función objetivo, pasando de valores iniciales cercanos a 180 000 a alrededor de 150 000 en un número reducido de iteraciones. Esta fase inicial corresponde a la exploración del espacio de soluciones, donde la selección natural y el operador de cruce permiten la rápida eliminación de soluciones de baja calidad y la combinación de características favorables de los individuos más aptos. A medida que aumenta el número de generaciones, la pendiente de las curvas disminuye gradualmente, lo que indica una transición a una fase de explotación, donde las mejoras se reducen cada vez más y el algoritmo comienza a estabilizarse en torno a las regiones prometedoras del espacio de soluciones. A partir de aproximadamente la generación 100, las tres ejecuciones muestran un comportamiento prácticamente estacionario, lo que indica que el algoritmo ha alcanzado un estado de convergencia. Aunque las trayectorias no son idénticas debido al componente estocástico introducido por las diferentes semillas, todas convergen a valores finales del mismo orden de magnitud, lo que refleja una buena estabilidad del método. La semilla 110 se destaca por lograr el mejor valor final, mientras que las otras dos corridas convergen a soluciones ligeramente superiores, sin diferencias drásticas, lo que confirma que el algoritmo no es demasiado sensible a la inicialización aleatoria.

	<b>semilla</b>	<b>objetivo</b>	<b>factible</b>	<b>cpu_time</b>
0	42	135240.274903	True	1.68750
1	110	131283.883814	True	2.00000
2	172	134740.141656	True	2.28125
	<b>estadístico</b>		<b>objetivo</b>	<b>cpu_time</b>
0	mejor		131283.883814	1.687500
1	peor		135240.274903	2.281250
2	media		133754.766791	1.989583
3	desviación estándar		2154.409509	0.297012

En cuanto a las tablas de resultados, la primera resume el rendimiento individual de las tres ejecuciones independientes. Se observa que las tres soluciones finales son factibles, lo que valida la correcta aplicación de las restricciones del modelo. Los valores de la función objetivo muestran variaciones moderadas entre ejecuciones: la ejecución con la semilla 110 obtiene el mejor valor con aproximadamente 131.284 unidades, mientras que la semilla 42 produce el peor valor con aproximadamente 135.240, y la semilla 172 se sitúa en un punto intermedio. Estas diferencias son consistentes con la naturaleza estocástica del algoritmo y también se reflejan en los tiempos de CPU, que oscilan entre aproximadamente 1,69 y 2,28 segundos, mostrando una baja dispersión y tiempos de ejecución estables entre ejecuciones. La segunda tabla consolida estos resultados mediante estadística descriptiva. Como era de esperar, el mejor valor corresponde a la semilla 110, mientras que el peor se asocia a la semilla 42. La función objetivo media ronda las 133.755 unidades, lo que representa un punto de referencia para el rendimiento promedio del algoritmo, y la desviación estándar de aproximadamente 2.154 confirma que la variabilidad entre ejecuciones es relativamente baja en comparación con la magnitud del objetivo. De igual forma, el tiempo de CPU tiene una media cercana a los 2 segundos y una desviación estándar baja, lo que indica que el coste computacional del algoritmo es consistente y predecible. En conjunto, estos resultados demuestran que el algoritmo genético exhibe un comportamiento robusto, con soluciones finales estables, tiempos de ejecución homogéneos y convergencia fiable a soluciones de alta calidad para el caso base.

### Caso 2 y 3:

Para los casos 2 y 3 se usaron las mismas semillas que en el caso base, pero no se llegó a un resultado favorable, a continuación se presentan los resultados:



Los gráficos de convergencia de los casos 2 y 3 muestran un comportamiento marcadamente diferente al del caso base. En ambos casos, las tres ejecuciones independientes presentan una caída extremadamente abrupta del valor de la función objetivo durante las primeras generaciones, seguida de una estabilización casi inmediata que permanece constante hasta el final del proceso evolutivo. Este patrón demuestra que el algoritmo genético identifica rápidamente una región de bajo coste en el espacio de soluciones, pero queda atrapado desde las primeras iteraciones en una solución que no mejora progresivamente con cada generación. A diferencia del caso base, donde la convergencia muestra una reducción gradual y sostenida, en los casos 2 y 3 la convergencia es prácticamente instantánea, lo que indica una pérdida temprana de diversidad en la población y la incapacidad de encontrar soluciones estructuralmente diferentes. Además, las tres semillas convergen exactamente al mismo valor en cada caso, lo que refleja que el problema, bajo las restricciones actuales, conduce de forma determinista al algoritmo a la misma configuración de minimización de costes, aunque esta configuración no cumpla con las restricciones de viabilidad.

<b>semilla</b>		<b>objetivo</b>	<b>factible</b>	<b>cpu_time</b>
0	42	7.342301e+06	False	1.062500
1	110	7.342301e+06	False	1.421875
2	172	7.342301e+06	False	0.921875

RESULTADOS CASO 2

<b>semilla</b>		<b>objetivo</b>	<b>factible</b>	<b>cpu_time</b>
0	42	1.379497e+07	False	11.750000
1	110	1.385218e+07	False	12.343750
2	172	1.379497e+07	False	12.234375

RESULTADOS CASO 3

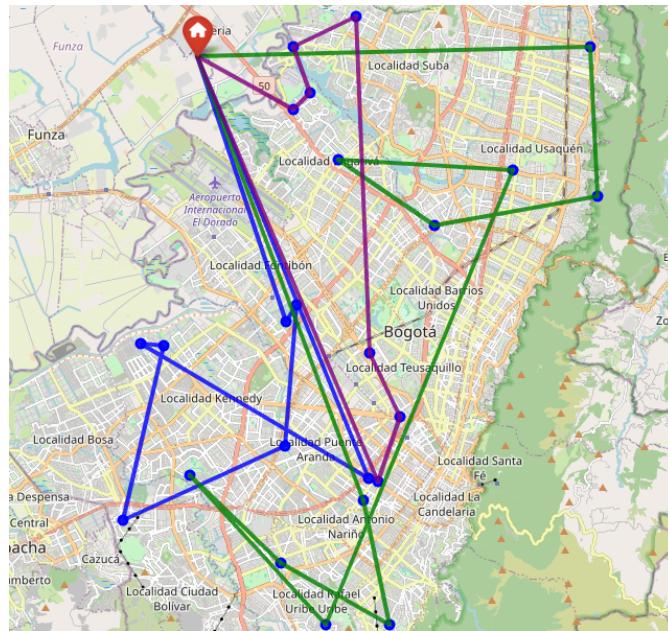
En cuanto a las tablas de resultados, se observa de forma consistente que, en los casos 2 y 3, ninguna de las tres ejecuciones produce una solución factible, independientemente del valor de semilla utilizado. Todos los valores finales se marcan como inviables, a pesar de que el algoritmo logra estabilizar un valor idéntico de la función objetivo en las tres ejecuciones. Este comportamiento se explica directamente por el contexto estructural del problema en esta versión, a diferencia de versiones anteriores, donde los casos 2 y 3 involucran múltiples depósitos, esta implementación solo permite el uso de una sola estación. Además, la limitada autonomía operativa de los vehículos impone una severa restricción a la cobertura geográfica y a la demanda total que se puede satisfacer. Bajo esta configuración, una sola estación es insuficiente para atender a todos los clientes dentro de los límites de capacidad y distancia permitidos, lo que imposibilita la construcción de rutas que cumplan simultáneamente todas las restricciones del modelo. Por esta razón, aunque el algoritmo logra minimizar el costo bajo las condiciones impuestas, todas las soluciones resultantes violan alguna restricción crítica del sistema, lo que se refleja en su inviabilidad. Mientras tanto, los tiempos de CPU se mantienen relativamente bajos y estables entre ejecuciones, lo que confirma que el problema no reside en la eficiencia computacional del algoritmo, sino en la inviabilidad estructural del escenario presentado con un solo depósito de combustible y baja autonomía vehicular. En conjunto, estos resultados confirman que los casos 2 y 3, en las condiciones actuales de esta versión, no admiten una solución factible, y que esta situación se debe a una limitación en la formulación logística del problema y no a un fallo del algoritmo genético como método de solución.

### c. Análisis comparativo con soluciones Pyomo

Caso Base:

Para recordar la solución realizada por Pyomo para el caso base, el resultado de la función objetivo fue de 436 865 pesos con un tiempo de ejecución de aproximadamente 3 minutos. Los resultados de la configuración fueron de la siguiente manera:

	VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime	FuelCost
0	V001	CD01	126.0	CD01-C004-C008-C005-C011-C017-C006-C018-CD01	7	15.0-20.0-20.0-17.0-25.0-17.0-12.0	94.648	229.216667	51425.413333
1	V002	CD01	138.0	CD01-C001-C010-C024-C009-C016-C019-C003-C002-C...	10	13.0-15.0-11.0-20.0-10.0-11.0-12.0-15.0-17.0-14.0	136.768	338.550000	74310.613333
2	V008	CD01	113.0	CD01-C015-C022-C013-C012-C020-C023-C014-CD01	7	17.0-18.0-21.0-12.0-15.0-15.0-15.0	75.622	168.366667	41087.953333



Para la solución por metaheurística, la mejor solución dada por la semilla de 110 obtuvo un valor objetivo de 131.283 pesos en un tiempo de 1,5 segundos, y los resultados fueron de la siguiente manera:

	VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime	Fuel
0	V001	CDA	42	CDA-C014-C002-C003-CDA	3	15-15-12	21.283406	0.472965	11563.98
1	V002	CDA	49	CDA-C009-C018-C006-CDA	3	20-12-17	32.311684	0.718037	17556.01
2	V003	CDA	55	CDA-C017-C001-C015-CDA	3	25-13-17	35.683292	0.792962	19387.92
3	V004	CDA	54	CDA-C012-C021-C007-C019-CDA	4	12-14-17-11	35.432725	0.787394	19251.78
4	V005	CDA	54	CDA-C013-C022-C004-CDA	3	21-18-15	34.203084	0.760069	18583.67
5	V006	CDA	40	CDA-C005-C008-CDA	2	20-20	22.100104	0.491113	12007.72
6	V007	CDA	53	CDA-C011-C016-C010-C024-CDA	4	17-10-15-11	50.980527	1.132901	27699.41
7	V008	CDA	30	CDA-C020-C022-CDA	2	15-15	9.631959	0.214044	5233.36



Para el caso base se realizó una comparación directa entre la solución obtenida mediante el modelo exacto implementado en Pyomo y la mejor solución encontrada por la metaheurística basada en algoritmo genético. En la Fase 2, Pyomo obtuvo un valor de la función objetivo de 436 865 pesos, con un tiempo de ejecución aproximado de 3 minutos, lo que refleja el alto costo computacional asociado a la búsqueda exacta de soluciones óptimas en este tipo de problemas combinatorios. En contraste, la metaheurística alcanzó su mejor solución con la semilla 110, obteniendo un valor objetivo de aproximadamente 131 283 pesos en un tiempo de apenas 1,5 segundos, lo que evidencia una mejora sustancial tanto en calidad de solución como en eficiencia computacional. Esta diferencia tan marcada en el valor de la función objetivo sugiere que, bajo la formulación y ponderación de costos utilizados en esta entrega, la metaheurística logró explorar de manera más efectiva configuraciones de ruteo de bajo costo, mientras que Pyomo quedó restringido por la complejidad del espacio de búsqueda.

Desde el punto de vista del uso de recursos operativos, también se observan diferencias significativas entre ambas soluciones. La solución de Pyomo utiliza únicamente 3 vehículos, concentrando la atención de la demanda en un número reducido de rutas de gran extensión. En cambio, la solución metaheurística emplea 7 vehículos, lo que permite un reparto más distribuido de la carga y un análisis más fino de las rutas. Esta diferencia en la cantidad de vehículos utilizados tiene un impacto directo sobre la longitud promedio de las rutas y sobre la configuración espacial del sistema. Las rutas de Pyomo presentan trayectos más largos y

continuos, con recorridos extensos que cubren múltiples zonas de la ciudad, mientras que las rutas obtenidas por la metaheurística son más cortas, fragmentadas y localizadas, como se evidencia en la visualización geográfica, donde se observan trayectorias más compactas que parten del depósito hacia sectores específicos.

En términos de balance de carga, la solución metaheurística muestra una distribución más homogénea entre los vehículos, evitando concentraciones excesivas de demanda en una sola ruta. Por el contrario, la solución de Pyomo, al emplear un número menor de vehículos, tiende a cargar de manera más intensiva a cada uno de ellos, lo que implica recorridos más largos y una mayor dependencia de la capacidad máxima de los vehículos. Esta diferencia también se refleja en las configuraciones de rutas: mientras Pyomo prioriza una estructura más centralizada con pocas rutas de gran cobertura, la metaheurística opta por una estrategia más descentralizada, con múltiples rutas de menor longitud y mayor especialización territorial.

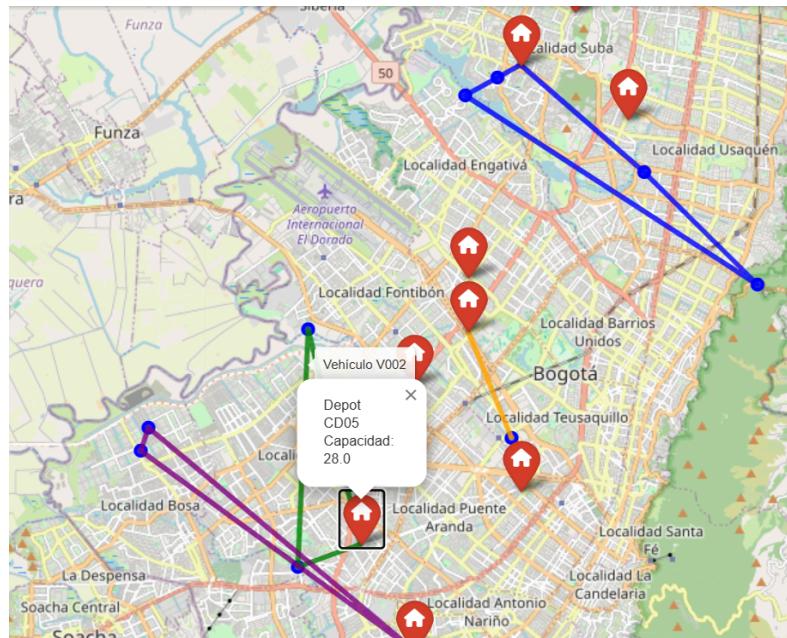
Desde el punto de vista computacional, la metaheurística presenta una ventaja clara en términos de tiempo de ejecución, al resolver el problema en segundos frente a los minutos requeridos por Pyomo. Esto la convierte en una alternativa especialmente atractiva para escenarios donde se requieren múltiples ejecuciones, análisis de sensibilidad o optimización frecuente. No obstante, debe tenerse en cuenta que la metaheurística no garantiza la optimización, sino soluciones de alta calidad en tiempos reducidos, mientras que Pyomo busca soluciones exactas bajo el marco del modelo matemático.

En conjunto, para el caso base, la comparación muestra que la metaheurística logra una mejor calidad de solución según el valor de la función objetivo, con un tiempo de cómputo extremadamente inferior, a costa de utilizar un mayor número de vehículos y de generar una configuración de rutas más fragmentada. Pyomo, por su parte, produce una solución con menos vehículos y una estructura más compacta, pero con un costo significativamente mayor y un tiempo de resolución considerablemente más alto. Esta diferencia ilustra claramente el equilibrio entre optimización exacta versus eficiencia computacional y flexibilidad operativa, y resalta el potencial de la metaheurística como herramienta práctica para problemas de ruteo urbano de gran escala.

## Caso 2:

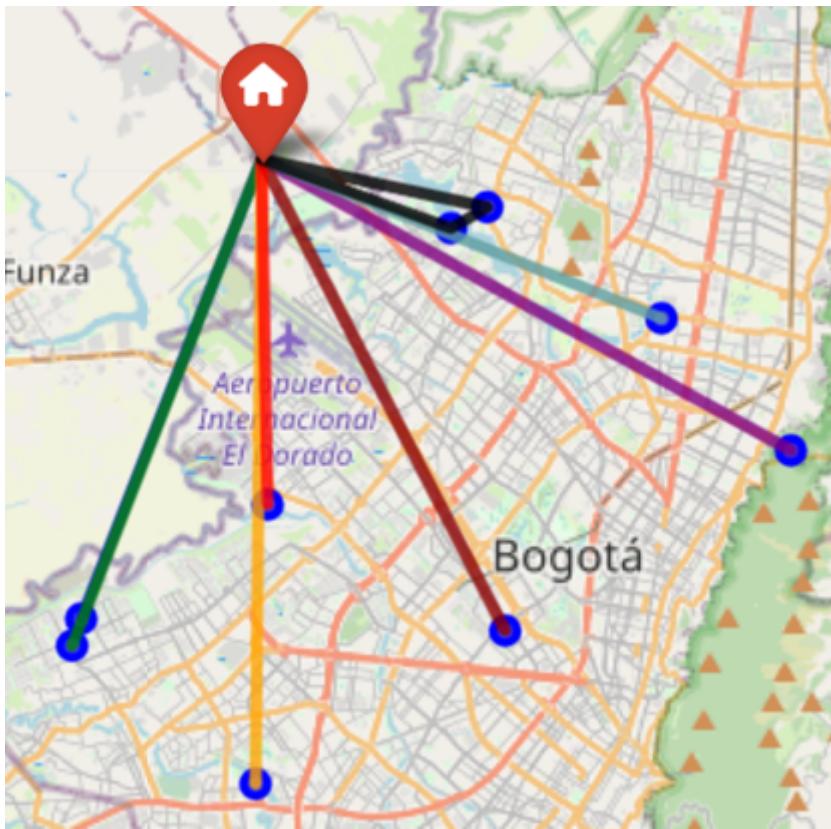
Para recordar la solución realizada por Pyomo para el caso 2, el resultado de la función objetivo fue de 558 702 pesos con un tiempo de ejecución de aproximadamente 6 minutos. Los resultados de la configuración fueron de la siguiente manera:

VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime	FuelCost	
0	V001	CD09	42.0	CD09-C008-C007-C005-C003-CD09	4	10.0-12.0-5.0-15.0	42.417	117.583333	17284.927500
1	V002	CD05	26.0	CD05-C006-C002-CD05	2	11.0-15.0	25.727	66.916667	13978.336667
2	V003	CD12	18.0	CD12-C004-C001-CD12	2	6.0-12.0	29.149	79.066667	15837.623333
3	V005	CD11	15.0	CD11-C009-CD11	1	15.0	10.427	25.666667	5665.336667



Para la solución por metaheurística, la solución dada, no factible, obtuvo un valor objetivo de 7 340 000 pesos en un tiempo de 1,2 segundos, y los resultados fueron de la siguiente manera:

VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime
0	V001	CDA	6	CDA-C004-CDA	1	6	25.030795 0.556240
1	V002	CDA	12	CDA-C001-CDA	1	12	26.457665 0.587948
2	V003	CDA	12	CDA-C007-CDA	1	12	30.623997 0.680533
3	V004	CDA	15	CDA-C002-CDA	1	15	31.587957 0.701955
4	V005	CDA	15	CDA-C009-CDA	1	15	26.842912 0.596509
5	V006	CDA	10	CDA-C008-CDA	1	10	21.740977 0.483133
6	V007	CDA	11	CDA-C006-CDA	1	11	17.427786 0.387284
7	V008	CDA	20	CDA-C005-C003-CDA	2	5-15	12.040430 0.267565



Para el caso 2, la comparación entre el modelo exacto implementado en Pyomo y la solución obtenida mediante la metaheurística evidencia diferencias aún más marcadas en términos de factibilidad del problema. En la Fase 2, cuando el caso 2 se resolvía utilizando múltiples depósitos, Pyomo logró encontrar una solución factible, con un valor de la función objetivo de 558 702 pesos y un tiempo de ejecución aproximado de 6 minutos. Esta solución hace uso de varios depósitos distribuidos geográficamente, lo que permite cubrir de manera eficiente la demanda total del sistema, reducir los recorridos excesivamente largos y respetar simultáneamente las restricciones de capacidad, autonomía y equilibrio de carga. La visualización de las rutas evidencia una estructura territorial coherente, donde cada vehículo cubre una zona relativamente compacta asociada a su depósito de origen.

En contraste, bajo las condiciones de esta entrega, donde el caso 2 fue forzado a operar con un único depósito, la metaheurística no logra encontrar ninguna solución factible en ninguna de las corridas realizadas. A pesar de que el algoritmo genético converge rápidamente a un mismo valor de la función objetivo en todas las semillas, todas las resultantes son infactibles, lo que confirma que el problema, bajo esta nueva restricción estructural, no admite solución. Esta situación se explica por la combinación de dos factores críticos: por un lado, la reducción del número de depósitos a uno solo, y por otro, la limitada autonomía operativa de los vehículos. Bajo estas condiciones, un único punto de salida resulta insuficiente para atender eficientemente a todos los clientes dispersos geográficamente, ya que los recorridos requeridos superan los límites de distancia, tiempo o capacidad permitidos por el modelo.

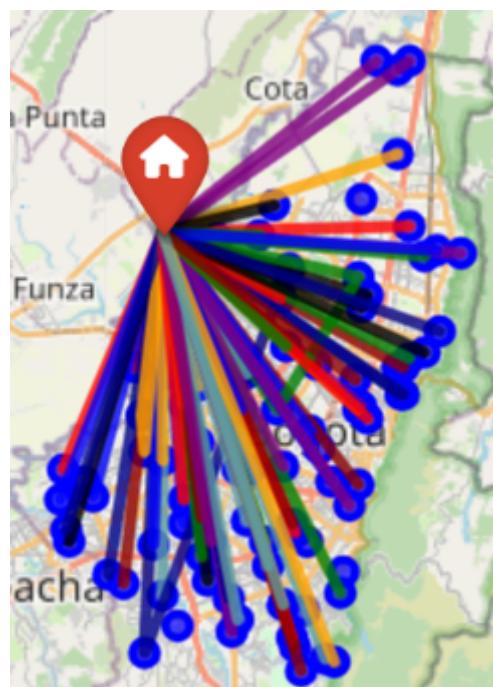
Desde el punto de vista cualitativo de las rutas, la solución de Pyomo presenta una organización descentralizada, con rutas relativamente balanceadas y recorridos adaptados

a la ubicación de los depósitos múltiples. La metaheurística, en cambio, al verse restringida a un único depósito, genera rutas que parten todas desde el mismo punto, provocando trayectorias excesivamente largas, superposición de recorridos y una alta concentración de esfuerzo logístico en un solo nodo. Esta configuración conduce inevitablemente a violaciones de las restricciones del sistema, lo que se refleja en la infactibilidad generalizada de las soluciones.

### Caso 3:

Para la solución por metaheurística, la solución dada, no factible, obtuvo un valor objetivo de 13 700 000 pesos en un tiempo de 11,75 segundos, y los resultados fueron de la siguiente manera:

VehicleId	DepotId	InitialLoad	RouteSequence	ClientsServed	DemandsSatisfied	TotalDistance	TotalTime
0	V001	CDA	12	CDA-C030-CDA	1	12	25.604710 0.568994
1	V002	CDA	12	CDA-C062-CDA	1	12	15.928472 0.353966
2	V003	CDA	12	CDA-C070-CDA	1	12	41.176167 0.915026
3	V004	CDA	25	CDA-C010-CDA	1	25	26.740494 0.594233
4	V005	CDA	12	CDA-C041-CDA	1	12	28.050722 0.623349
...	...	...	...	...	...	...	...
81	V082	CDA	12	CDA-C078-CDA	1	12	16.877899 0.375064
82	V083	CDA	12	CDA-C045-CDA	1	12	31.840796 0.707573
83	V084	CDA	19	CDA-C086-CDA	1	19	46.915008 1.042556
84	V085	CDA	12	CDA-C069-CDA	1	12	24.432568 0.542946
85	V086	CDA	12	CDA-C037-CDA	1	12	35.376933 0.786154



Para el caso 3 no se cuenta con una solución concreta obtenida mediante Pyomo en la fase anterior, ya que el modelo exacto presentó tiempos de cómputo excesivamente altos, lo que impidió alcanzar una solución dentro de un horizonte de tiempo razonable. Esto ya muestra la alta complejidad computacional de esta instancia, asociada principalmente al elevado

número de clientes y a la estructura del problema. En la implementación actual mediante metaheurística, el algoritmo genético logró producir una solución en un tiempo de 11,75 segundos, con un valor de la función objetivo cercano a 13 700 000 pesos, sin embargo, todas las corridas resultaron infactibles. La tabla de resultados y la visualización de rutas muestran que la solución alcanza a utilizar hasta 86 vehículos, todos partiendo desde un único depósito, lo que evidencia una configuración altamente fragmentada y operativamente poco realista. Este comportamiento confirma que, bajo la restricción de un solo depósito y la autonomía limitada de los vehículos, el caso 3 no admite una solución viable, igual a lo observado en el caso 2, pero con un impacto mucho mayor debido a la escalada del problema. Con esto, la comparación con Pyomo refuerza la conclusión de que no solo la metaheurística enfrenta serias dificultades computacionales cuando el problema no está estructuralmente bien condicionado, ya sea por una mala distribución de depósitos o por el tamaño extremo de la instancia.

## 4. Análisis de Escalabilidad

### a. Rendimiento en instancias de diferentes tamaños

En el caso base, la metaheurística produce valores cercanos a los obtenidos por Pyomo. En el caso mediano, empieza a observarse una diferencia más notoria entre ambas soluciones. Finalmente, en el caso grande, el aumento en la cantidad de clientes amplía considerablemente el espacio de búsqueda, lo que provoca que los valores obtenidos por la metaheurística se alejen aún más de los resultados del modelo exacto. Esto confirma que, a mayor tamaño de la instancia, mayor es la pérdida de precisión frente a Pyomo. Sin embargo, la velocidad de ejecución es mucho mayor con el algoritmo genético utilizado que con cualquier solver de pyomo

### b. Límites prácticos de aplicabilidad

El principal límite práctico del método aparece en instancias grandes, donde el incremento en la cantidad de datos reduce la capacidad de la metaheurística para aproximarse al óptimo global. Además, si la demanda total supera la capacidad total de la flota, el problema se vuelve infactible desde el punto de vista matemático, independientemente del algoritmo utilizado. Esto se debe a los cambios de las restricciones y los datos usados en los casos 2 y 3, con las capacidades de los vehículos de distancia y carga, es imposible para los vehículos recorrer toda la ciudad desde un único centro de distribución, por lo que las soluciones del algoritmo usualmente rompen la restricción de “range” y resulta en una solución infactible.

### c. Estrategias para mejorar la escalabilidad

Para reducir la brecha entre la metaheurística y Pyomo en instancias grandes, se pueden mejorar las heurísticas de inicialización, aumentar el número de iteraciones, ajustar mejor los parámetros y combinar la metaheurística con búsqueda local. Estas estrategias permiten mejorar la calidad de las soluciones sin incrementar de forma excesiva el tiempo de ejecución.

## **5. Discusión**

### **a. Ventajas y desventajas de cada enfoque**

En cuanto a las ventajas y desventajas de cada enfoque, los métodos exactos como Pyomo se destacan por ofrecer optimidad garantizada y soluciones completamente verificables, lo cual resulta fundamental en problemas donde la precisión matemática es prioritaria y las instancias son de tamaño moderado. Sin embargo, su principal limitación es el alto costo computacional, que crece de manera exponencial con el número de clientes y restricciones, haciendo que su uso sea poco viable en escenarios de gran escala. Por otro lado, las metaheurísticas presentan como principales fortalezas su alta escalabilidad, su flexibilidad para adaptarse a distintos ajustes del problema y sus tiempos de cómputo reducidos. Sin embargo, estas no garantizan optimalidad, sino soluciones de alta calidad. En este sentido, el equilibrio entre ambos enfoques queda claramente definido, mientras los métodos Pyomo priorizan la precisión, las metaheurísticas priorizan la eficiencia computacional y la aplicabilidad práctica.

### **b. Recomendaciones para diferentes escenarios**

Desde una perspectiva aplicada, la elección entre un método exacto y una metaheurística depende del contexto operativo. En escenarios reales con pocos clientes, estructuras bien definidas y donde se requiere una solución óptima garantizada, el uso de Pyomo es altamente recomendable. En cambio, en contextos con grandes volúmenes de clientes, múltiples restricciones operativas, necesidad de optimización frecuente o toma de decisiones en tiempo real, las metaheurísticas se presentan como la alternativa más adecuada; pues su capacidad para generar soluciones rápidas y suficientemente buenas las convierte en herramientas ideales para sistemas logísticos dinámicos, donde el tiempo de respuesta es un factor crítico.

### **c. Lecciones aprendidas y desafíos encontrados**

Las principales lecciones aprendidas muestran la importancia del diseño estructural del problema, particularmente en lo relacionado con la cantidad y ubicación de los depósitos, ya que una mala configuración puede hacer que el problema no admita soluciones factibles. También, se evidencia que la metaheurística puede ser altamente efectiva cuando el problema está bien planteado, pero también es sensible a restricciones demasiado estrictas, como la reducción forzada a un solo depósito. Entre los desafíos más relevantes se encuentran la calibración adecuada de los parámetros del algoritmo genético, la gestión de la infactibilidad en instancias grandes y la alta demanda computacional de los métodos exactos en escenarios complejos. Estos aspectos reafirman la necesidad de un enfoque equilibrado entre modelación matemática rigurosa y técnicas heurísticas de resolución.

## **6. Conclusiones**

### **a. Resumen de hallazgos principales**

El estudio realizado evidencia diferencias claras entre el desempeño del modelo exacto y la metaheurística basada en algoritmo genético (GA) para resolver el CVRP bajo diversas configuraciones de instancia. Para el caso base, muestra un rendimiento muy bueno, llegando a valores menores que los obtenidos en pyomo, y en tiempos de ejecución más cortos.

Por el contrario, las soluciones del caso 2 y 3 no llegaron a ser factibles con la restricción de un único depósito. Todas las ejecuciones resultaban inviables debido a la insuficiente capacidad operativa de la flota para cubrir toda la demanda, y cumplir con el rango desde un solo depósito.

En cuanto a escalabilidad, la metaheurística provee gran eficiencia para instancias pequeñas y medianas, pero disminuye su capacidad de acercarse al óptimo al aumentar el tamaño del problema. Aún así los tiempos siguen siendo menores a los vistos en pyomo, y puede ser usada para momentos donde no es necesaria la exactitud en encontrar el valor óptimo.

### **b. Respuestas a preguntas estratégicas**

1. ¿El uso del Algoritmo Genético para resolver el problema CVRP?

Sí, debido a que los tiempos de respuesta vistos en la solución son muy buenos, y obtiene resultados de buena calidad y estabilidad. Para instancias más grandes, depende de los parámetros dados, requiriendo de mayor escalabilidad.

2. ¿Por qué los casos 2 y 3 son inviables al usar el Algoritmo Genético?

Esto se debe a que el uso de un depósito puede ser utilizado en un instancia más pequeña como lo es en el caso base, pero a medida que aumenta se requiere de más depósitos para que los vehículos no excedan su rango. Otra manera para solucionarlo, sería ampliando las capacidades y rangos de los vehículos, para así poder satisfacer la demanda de los clientes.

3. ¿Es recomendable el uso del Algoritmo Genético con respecto a métodos más exactos?

Sí. Mientras que Pyomo garantiza la optimalidad, su costo computacional crece rápidamente con el tamaño del problema. Por el contrario, la metaheurística ofrece soluciones aproximadas de alta calidad en segundos, lo que la convierte en la opción más adecuada para escenarios operativos reales con muchas variables, necesidad de rápida toma de decisiones o múltiples ejecuciones sucesivas.

4. ¿Qué factores llegan a afectar la calidad de la solución?

La estructura de la instancia. Cómo se llegó a observar en los casos 2 y 3, que usaban valores factibles en el caso base, pero no en dichos casos. Otro sería la correcta elección o calibración de los parámetros en el GA (como el tamaño de población, probabilidades de cruce y mutación). Otro factor que probablemente llega a afectar la solución es la diversidad encontrada en la población inicial, ya que una población inicial poco diversa impide la exploración de resultados y reduce la calidad del cruce.

### **c. Direcciones futuras de investigación**

Para investigaciones futuras, se recomienda:

- La incorporación de múltiples depósitos para problemas más grandes. Como se pudo evidenciar en los casos 2 y 3, utilizar un único depósito no ayuda a la escalabilidad, y se requiere ampliar dicha cantidad para el funcionamiento en instancias más grandes.
- Además, se podría incluir mayor cantidad de restricciones para un simulación más acertada y realista, como puede ser la inclusión de peajes o repostaje de combustible.
- Se podría combinar con otros algoritmos heurísticos como 2-opt, para refinar y mejorar la descendencia generada, y así mejorar la explotación del espacio de búsqueda y aumentar la capacidad del algoritmo.
- Por último, debería hacerse un análisis de sensibilidad para estudiar la variación en el objetivo que se genera al hacer un cambio en los parámetros como aumentar la capacidad, demanda, depósitos o incluso disminuirlos.