

Proyecto Etapa III Implementación de Metaheurísticas:

Optimización en logística para la empresa LogistiCo

Curso: ISIS-3302
Departamento de Ingeniería de Sistemas
Universidad de los Andes

1 Introducción

Este documento presenta la tercera etapa del proyecto de optimización en logística para LogistiCo. Esta etapa se enfoca en la implementación de un método metaheurístico para resolver exclusivamente el problema base del CVRP (Capacitated Vehicle Routing Problem) y comparar los resultados con las soluciones obtenidas mediante Pyomo en la fase anterior. El objetivo es analizar las ventajas y desventajas de cada enfoque en términos de calidad de solución, tiempo de ejecución y escalabilidad.

2 Material de Apoyo Proporcionado

Para facilitar la implementación de esta fase, se proporciona un **ejemplo completo y bien documentado de un Algoritmo Genético (GA)** resolviendo el **Problema del Viajante (TSP)**. Este ejemplo incluye:

- Implementación modular y comentada de un GA para TSP
- Explicación detallada de:
 - Representación de soluciones (cromosomas como permutaciones)
 - Operadores de cruce (PMX, OX, etc.)
 - Operadores de mutación (swap, inversión)
 - Función de evaluación

-
- Selección y reemplazo
 - Código Python reutilizable y extensible
 - Ejemplos de visualización de convergencia y rutas

Se espera que los estudiantes utilicen este ejemplo como base para desarrollar su solución al CVRP. La tarea principal consiste en:

1. **Comprender** el código del ejemplo de GA para TSP
2. **Adaptar** la representación y operadores para manejar múltiples rutas (CVRP)
3. **Incorporar** el manejo de la restricción de capacidad
4. **Modificar** la función de evaluación para incluir costos del CVRP

Nota importante: Si bien se proporciona un ejemplo de GA, los estudiantes pueden optar por implementar otra metaheurística (Búsqueda Tabú, Recocido Simulado, ACO, etc.), pero en ese caso deben desarrollarla desde cero sin material de apoyo adicional.

3 Evaluación de los entregables

Para la entrega correspondiente a esta etapa, los estudiantes deberán presentar un documento en formato PDF o *Jupyter Notebook* que incluya el desarrollo completo de la implementación de un método metaheurístico para resolver el CVRP base, así como la comparación con las soluciones obtenidas mediante Pyomo. Los entregables se evaluarán conforme a los siguientes criterios:

1. **Implementación de Metaheurísticas (35 %):**
 - Correcta implementación de un algoritmo metaheurístico para resolver el CVRP base.
 - Adecuada parametrización y calibración del algoritmo implementado.
 - Correcto manejo de las restricciones del problema.
2. **Comparación con Soluciones Previas (25 %):**
 - Análisis comparativo detallado entre las soluciones obtenidas mediante la metaheurística y las obtenidas con Pyomo.

-
- Evaluación de calidad de solución (valor de función objetivo), tiempo de ejecución y recursos computacionales.
 - Discusión sobre ventajas y desventajas de cada enfoque.

3. Análisis de Escalabilidad (20 %):

- Evaluación de la escalabilidad del método metaheurístico implementado para instancias de mayor tamaño.
- Análisis de la degradación de rendimiento a medida que aumenta el tamaño del problema.
- Conclusiones sobre la aplicabilidad de cada enfoque para problemas reales de gran escala.

4. Visualización y Evaluación (10 %):

- Visualizaciones adecuadas para comparar las rutas obtenidas por el método metaheurístico y Pyomo.
- Gráficos de convergencia y evolución de las soluciones para el método metaheurístico.

5. Documentación y Calidad del Código (10 %):

- Código limpio, bien estructurado y adecuadamente comentado.
- Modularidad y reutilización de componentes.
- Manejo efectivo de excepciones y casos límite.
- Documentación clara que permita comprender la implementación.

4 Instrucciones

1. Ámbito del problema: CVRP Simplificado

Problema a resolver: Se implementará el *Capacitated Vehicle Routing Problem (CVRP) básico* utilizando metaheurísticas. Esta fase se enfoca en el problema fundamental, **ignorando las restricciones especiales** del Proyecto A de las fases anteriores.

Restricciones que SÍ se deben considerar:

- **Capacidad de vehículos:** Los vehículos tienen capacidad máxima Q
- **Unicidad de visita:** Cada cliente debe ser visitado exactamente una vez
- **Origen y destino:** Todas las rutas inician y terminan en el depósito
- **Demanda de clientes:** Cada cliente tiene una demanda que debe satisfacerse
- **Función objetivo:** Utilizar la misma función objetivo de la Fase 2 (costos fijos, distancia, tiempo, combustible)

Restricciones del Proyecto A que SE IGNORAN:

- **NO** usar múltiples depósitos (usar solo el del Caso Base)
- **NO** considerar capacidad de inventario de depósitos
- **NO** considerar restricciones de zonas urbanas
- **NO** diferenciar entre tipos de vehículos (considerar flota homogénea)

Instancias a ejecutar:

Se utilizarán las tres instancias del **Proyecto A**, cada una con diferente escala:

Instancia	Descripción
Proyecto_Caso_Base	Caso pequeño
Proyecto_A_Caso2	Caso mediano
Proyecto_A_Caso3	Caso grande

Cada instancia corresponde a un subconjunto de clientes con distinta escala y se almacena en su propio directorio de datos.

Archivos a utilizar por instancia:

- `clients.csv`: Usar columnas `ClientID`, `StandardizedID`, `LocationID`, `Latitude`, `Longitude`, `Demand`
 - Ignorar otras columnas si existen
- `vehicles.csv`: Usar columnas `VehicleID`, `StandardizedID`, `Capacity`, `Range`
 - Si hay vehículos con diferentes capacidades, considerar flota homogénea usando la capacidad más común o un promedio representativo
- `depots.csv`: Usar **ÚNICAMENTE** el depósito del Proyecto `_Caso_Base`
 - Solo necesita las coordenadas (`Latitude`, `Longitude`) del depósito
 - Si los archivos de `Caso2/Caso3` tienen múltiples depósitos, ignorarlos
 - **NO considerar columna `Capacity` de depósitos**
- `parameters_*.csv`: Utilizar los parámetros de costos de la Fase 2
 - Costos fijos por vehículo (`C_fixed`)
 - Costos por distancia (`C_dist`)
 - Costos por tiempo (`C_time`)
 - Costos de combustible y eficiencia

Función objetivo:

Utilizar la **misma función objetivo de la Fase 2**:

Objetivo: Comparar la *escalabilidad* del método metaheurístico frente al modelo exacto (Pyomo) cuando crece el número de clientes.

2. Organización del repositorio

- Conserve la estructura de directorios establecida en fases anteriores.
- Actualice el `README.md` detallando:
 - Descripción del metaheurístico seleccionado.
 - Cómo reproducir los experimentos (rutas de datos, comandos, semillas, etc.).

3. Implementación del método metaheurístico

- Seleccione y programe **un** método entre: *Algoritmo Genético*, *Búsqueda Tabú*, *Recocido Simulado*, *Optimización por Colonia de Hormigas*, *Búsqueda Local Iterativa* u otro (con aprobación previa).
- **Se recomienda fuertemente usar Algoritmo Genético**, ya que se proporciona un ejemplo completo de GA para TSP que puede adaptarse.
- Desarrolle operadores específicos para CVRP:
 - Representación de soluciones (cromosoma, lista de rutas, etc.).
 - Operadores de cruce, mutación y/o movimiento tabú.
 - Heurística de reparación o penalización para mantener la factibilidad (respeto de capacidad Q y unicidad de visita).

3.1 Adaptación del ejemplo de GA para TSP a CVRP

El ejemplo proporcionado de GA para TSP resuelve un problema con una sola ruta. Para adaptarlo a CVRP con múltiples vehículos, considere las siguientes modificaciones:

- **Representación:**
 - TSP: Cromosoma = permutación de clientes $[c_1, c_2, \dots, c_n]$
 - CVRP: Agregar delimitadores entre rutas, por ejemplo: $[c_1, c_2, |, c_3, c_4, c_5, |, c_6, \dots]$
 - Alternativa: Usar lista de listas $[[c_1, c_2], [c_3, c_4, c_5], [c_6, \dots]]$
- **Función de evaluación:**
 - TSP: Suma de distancias en una ruta circular
 - CVRP: Suma de distancias de múltiples rutas + costos fijos + costos de tiempo + costos de combustible
 - Agregar penalización si se viola la capacidad del vehículo
- **Operadores genéticos:**
 - Los operadores de cruce y mutación del TSP pueden reutilizarse
 - Agregar operadores específicos para CVRP: mover cliente entre rutas, fusionar rutas, dividir rutas

- **Reparación de soluciones:**

- Si una ruta excede capacidad: dividirla o reasignar clientes
- Implementar heurística de inserción para mantener factibilidad

Ejemplo conceptual de transformación:

```
1 # TSP: Una ruta
2 ruta_tsp = [0, 3, 1, 4, 2, 0] # Deposito 0, clientes 1-4
3
4 # CVRP: Múltiples rutas con capacidad Q=100
5 # Cliente 1: 40kg, Cliente 2: 50kg, Cliente 3: 60kg, Cliente 4: 30kg
6 rutas_cvrp = [
7     [0, 3, 4, 0], # Ruta 1: 60+30=90kg (factible)
8     [0, 2, 1, 0] # Ruta 2: 50+40=90kg (factible)
9 ]
```

4. Calibración y experimentación

- Diseñe un plan experimental para sintonizar los parámetros del algoritmo.
- Ejecute al menos **3 corridas independientes** por configuración, usando semillas distintas.
- Registre tiempo de CPU, valor de la función objetivo y estadísticas relevantes (mejor, peor, media, desviación estándar).

5. Análisis comparativo con soluciones de Pyomo

Compare su implementación metaheurística con los resultados obtenidos en la **Fase 2**:

- **Para las tres instancias (Caso Base, A_Caso2, A_Caso3):**
 1. Comparación de calidad de solución: Valor de la función objetivo (metaheurística vs Pyomo)
 2. Tiempo de ejecución
 3. Uso de memoria
 4. Comportamiento de escalabilidad al aumentar el número de clientes
- **Análisis cualitativo de las rutas:**

-
- Número de vehículos utilizados
 - Longitud promedio de las rutas
 - Balance de carga entre vehículos
 - Diferencias en las configuraciones de rutas

■ **Discusión esperada:**

- Ventajas de métodos exactos (Pyomo): Optimalidad garantizada, soluciones verificables
- Ventajas de metaheurísticas: Escalabilidad, flexibilidad, tiempos razonables para instancias grandes
- Trade-offs entre calidad y tiempo
- Recomendaciones sobre cuándo usar cada enfoque en contextos reales

Nota importante: Para esta comparación, use los resultados de Pyomo de la Fase 2 con la función objetivo completa (misma que utilizará en esta fase). Si en la Fase 2 sus instancias tenían restricciones adicionales del Proyecto A (múltiples depósitos, capacidad de inventario), puede re-ejecutar Pyomo con el modelo simplificado o hacer una comparación cualitativa explicando las diferencias.

6. Visualización de resultados

- Grafique la evolución de la mejor solución (*curva de convergencia*) para el metaheurístico.
- Trace las rutas finales superpuestas en el plano para metaheurístico y Pyomo.
- Presente histogramas o cajas de distribución de cargas y longitudes de ruta.

7. Archivos de verificación

- Genere un archivo de verificación por cada instancia ejecutada:
`verificacion_metaheuristica_<metodo>_<instancia>.csv`
- Ejemplos:
 - `verificacion_metaheuristica_GA_caso_base.csv`
 - `verificacion_metaheuristica_GA_A_caso2.csv`

-
- `verificacion_metaheuristica_GA_A_caso3.csv`
 - Use el mismo formato de columnas que en la Fase 2 para permitir comparación directa:
 - **VehicleId**: Identificador del vehículo (usar `StandardizedID`: V001, V002, etc.)
 - **DepotId**: Identificador del depósito (usar el del caso base)
 - **InitialLoad**: Carga inicial del vehículo (kg)
 - **RouteSequence**: Secuencia de la ruta (ej: CD01-C005-C023-C017-CD01)
 - **ClientsServed**: Número de clientes servidos en la ruta
 - **DemandsSatisfied**: Demandas satisfechas (separadas por guiones)
 - **TotalDistance**: Distancia total de la ruta (km)
 - **TotalTime**: Tiempo total de la ruta (horas)
 - **FuelCost**: Costo de combustible de la ruta (COP)
 - **TotalCost**: Costo total de la ruta incluyendo todos los componentes (COP)

8. Entregables

1. Código fuente:

- Implementación de la metaheurística (basada o inspirada en el ejemplo de GA para TSP)
- Código modular, comentado y ejecutable
- Scripts para reproducir los experimentos
- Estructura organizada en módulos (representación, evaluación, operadores, algoritmo, experimentación, visualización)

2. Archivos de verificación: Generar para las tres instancias:

- `verificacion_metaheuristica_<metodo>_caso_base.csv`
- `verificacion_metaheuristica_<metodo>_A_caso2.csv`
- `verificacion_metaheuristica_<metodo>_A_caso3.csv`
- Formato: Mismas columnas que la Fase 2 (detalladas en sección 7)

3. Visualizaciones:

-
- Mapas de rutas finales (metaheurística vs Pyomo) para cada instancia
 - Gráficos de convergencia del algoritmo metaheurístico
 - Gráficos comparativos de tiempo vs calidad
 - Histogramas de distribución de cargas y longitudes de ruta

4. Informe técnico (PDF o Jupyter Notebook):

- Descripción de las adaptaciones realizadas al ejemplo base (si aplica)
- Explicación de la representación de soluciones y operadores implementados
- Proceso de calibración de parámetros
- Análisis comparativo con resultados de Pyomo (Fase 2)
- Análisis de escalabilidad
- Discusión de ventajas y desventajas de cada enfoque
- Conclusiones y recomendaciones

5. README actualizado:

- Instrucciones para ejecutar el código
- Descripción de la estructura del proyecto
- Explicación de las modificaciones al ejemplo de GA para TSP (si aplica)
- Dependencias y requisitos (librerías, versiones de Python, etc.)
- Nombres de los integrantes del grupo

Formato de entrega: Repositorio Git con estructura organizada o archivo comprimido (.zip) subido a Bloque Neón.

5 Consideraciones para la Implementación de Metaheurísticas

5.1 Algoritmos Recomendados

Se recomienda seleccionar alguno de los siguientes algoritmos metaheurísticos para implementar:

- **Algoritmo Genético (GA):**

- Desarrollar una representación adecuada para rutas de vehículos
- Implementar operadores de cruce específicos para CVRP.
- Diseñar operadores de mutación que preserven la factibilidad
- Implementar mecanismos de reparación para soluciones no factibles

- **Búsqueda Tabú (TS):**

- Definir movimientos específicos para el CVRP (e.g., 2-opt, intercambio de clientes)
- Implementar estructuras de memoria a corto y largo plazo
- Desarrollar criterios de aspiración
- Diseñar estrategias de diversificación e intensificación

- **Optimización por Colonia de Hormigas (ACO):**

- Diseñar estrategias de construcción de rutas
- Implementar actualización de feromonas específica para CVRP
- Definir heurísticas locales para guiar la búsqueda
- Desarrollar mecanismos de balance entre exploración y explotación

5.2 Estructura del Código

Se recomienda seguir una estructura modular para la implementación de los algoritmos:

- **Módulo de representación:** Clases y estructuras para representar soluciones.
- **Módulo de evaluación:** Funciones para calcular la función objetivo y verificar restricciones.
- **Módulo de operadores:** Implementación de operadores específicos para cada metaheurística.
- **Módulo de algoritmo:** Implementación principal de cada metaheurística.
- **Módulo de experimentación:** Código para ejecutar experimentos, recopilar estadísticas y generar gráficos.

-
- **Módulo de visualización:** Funciones para visualizar soluciones y resultados comparativos.

6 Análisis Comparativo

Para realizar un análisis comparativo completo entre las soluciones obtenidas mediante Pyomo y las generadas por los métodos metaheurísticos, se sugiere considerar los siguientes aspectos:

6.1 Métricas de Calidad

Evaluar la calidad de las soluciones utilizando las siguientes métricas:

- **Valor de la función objetivo:** Comparar el costo total obtenido por cada método.
- **GAP de optimalidad:** Calcular la diferencia porcentual respecto a la solución óptima (o mejor conocida).
- **Factibilidad:** Verificar el cumplimiento de todas las restricciones.
- **Distribución de carga:** Analizar el balance de carga entre vehículos.
- **Distancia total recorrida:** Comparar la eficiencia de las rutas.

6.2 Métricas de Rendimiento

Evaluar el rendimiento computacional utilizando:

- **Tiempo de ejecución:** Medir el tiempo total requerido por cada método.
- **Memoria utilizada:** Cuantificar los recursos de memoria requeridos.
- **Escalabilidad:** Analizar cómo varía el rendimiento al aumentar el tamaño del problema.
- **Convergencia:** Estudiar la velocidad de convergencia de los métodos metaheurísticos.
- **Robustez:** Evaluar la consistencia de los resultados en múltiples ejecuciones.

6.3 Visualización Comparativa

Desarrollar las siguientes visualizaciones para facilitar la comparación:

- **Mapas comparativos:** Mostrar las rutas generadas por cada método.
- **Gráficos de convergencia:(Solo para las Metaheurísticas)** Visualizar la evolución de la función objetivo.
- **Gráficos de tiempo vs. calidad:** Analizar el compromiso entre tiempo y calidad.

7 Consideraciones Adicionales

7.1 Medidas de Complejidad

Es importante discutir las siguientes consideraciones de complejidad:

- **Complejidad teórica:** Analizar la complejidad algorítmica de los métodos implementados.
- **Complejidad práctica:** Evaluar el comportamiento real en términos de tiempo y memoria.
- **Escalabilidad:** Determinar límites prácticos para cada enfoque.
- **Complejidad de implementación:** Discutir el esfuerzo requerido para implementar y mantener cada método.

7.2 Calibración de Parámetros

Abordar los siguientes aspectos relacionados con la calibración:

- **Metodología de calibración:** Describir el enfoque utilizado (e.g., diseño de experimentos, optimización de parámetros).
- **Sensibilidad:** Analizar la sensibilidad de los resultados a cambios en los parámetros.
- **Parámetros críticos:** Identificar los parámetros que más impactan el rendimiento.
- **Configuraciones recomendadas:** Proporcionar valores recomendados para diferentes escenarios.

8 Estructura del Informe Final

El informe final debe seguir la siguiente estructura:

1. Descripción del Problema

- Formulación matemática del CVRP
- Características de la instancia base
- Restricciones y consideraciones

2. Método Implementado

- Descripción detallada del método metaheurístico implementado
- Estrategias de representación y operadores
- Proceso de calibración de parámetros

3. Resultados Experimentales

- Configuración experimental
- Presentación de resultados por método
- Análisis comparativo con soluciones Pyomo
- Visualizaciones y gráficos relevantes

4. Análisis de Escalabilidad

- Rendimiento en instancias de diferentes tamaños
- Límites prácticos de aplicabilidad
- Estrategias para mejorar la escalabilidad

5. Discusión

- Ventajas y desventajas de cada enfoque
- Recomendaciones para diferentes escenarios
- Lecciones aprendidas y desafíos encontrados

6. Conclusiones

- Resumen de hallazgos principales
- Respuestas a preguntas estratégicas
- Direcciones futuras de investigación

9 Consideraciones para la Entrega

- **Fecha límite:** La entrega debe realizarse a través de Bloque Neón antes del **Sábado 6 de Diciembre de 2025**
- **Formato:** El informe debe entregarse en formato PDF o como Jupyter Notebook.
- **Código fuente:** Todo el código fuente debe estar adecuadamente comentado y organizado.
- **Verificación:** Incluir los archivos de verificación para las tres instancias (Caso Base, A_Caso2, A_Caso3).
- **Grupos:** Esta actividad se puede realizar en grupos de hasta 4 integrantes.
- **Importante:** Adjuntar los nombres de los integrantes del equipo en el README y en el informe.

10 Bibliografía Recomendada

Nota: Dado que se proporciona un ejemplo completo de implementación de GA para TSP, la siguiente bibliografía es **opcional y complementaria** para estudiantes que deseen profundizar en la teoría o explorar otras metaheurísticas.

Se recomienda consultar las siguientes referencias:

1. Gendreau, M., & Potvin, J. Y. (Eds.). (2010). Handbook of metaheuristics (Vol. 146). Springer Science & Business Media.
2. Toth, P., & Vigo, D. (Eds.). (2014). Vehicle routing: problems, methods, and applications. Society for Industrial and Applied Mathematics.
3. Laporte, G. (2009). Fifty years of vehicle routing. Transportation Science, 43(4), 408-416.
4. Golden, B. L., Raghavan, S., & Wasil, E. A. (Eds.). (2008). The vehicle routing problem: latest advances and new challenges (Vol. 43). Springer Science & Business Media.

-
5. Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J. Y., & Semet, F. (2002). A guide to vehicle routing heuristics. *Journal of the Operational Research Society*, 53(5), 512-522.
 6. Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12), 1985-2002.
 7. Cordeau, J. F., & Laporte, G. (2005). Tabu search heuristics for the vehicle routing problem. In *Metaheuristic Optimization via Memory and Evolution* (pp. 145-163). Springer.
 8. Bell, J. E., & McMullen, P. R. (2004). Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics*, 18(1), 41-48.
 9. Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41(4), 421-451.
 10. Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2013). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research*, 231(1), 1-21.