Proyecto Seneca-Libre: Etapa 2 & 3 - Implementación y Resolución del Modelo de Ruteo Vehicular

Instructores:
Carlos Andrés Lozano,
Germán Adolfo Montoya,
Juan Andrés Mendez

Abstract

Este documento aborda las etapas dos y tres del proyecto de optimización logística de Seneca Libre, centradas en la implementación y validación del modelo matemático desarrollado para mejorar la eficiencia en las rutas de suministro. El objetivo principal es comprobar que el modelo refleja de manera precisa las operaciones logísticas reales y permite predecir rutas y costos de forma confiable. Para ello, se implementará el modelo en entornos de programación y se resolverán escenarios de prueba representativos, generando datos que validen su viabilidad y precisión.

Además, se realizará un análisis exhaustivo de los resultados, identificando oportunidades de mejora en la eficiencia operativa y reducción de costos. Con esta validación, Seneca Libre busca consolidarse como líder en el comercio electrónico, sentando las bases para simulaciones futuras y ajustes estratégicos que fortalezcan la sostenibilidad y competitividad de sus operaciones.

1 Introducción

Seneca Libre se posiciona como líder en el comercio electrónico en Bogotá, comprometido con una operación logística eficiente y competitiva. En la primera etapa de este proyecto, desarrollamos un modelo matemático integral diseñado para optimizar las rutas de suministro, enfrentando desafíos como alta demanda, costos elevados y las limitaciones de una flota compuesta por más de 700 vehículos. Este modelo se enfocó en reducir costos operativos, mejorar la puntualidad de las entregas y equilibrar la carga de trabajo de los transportistas.

En esta segunda fase, el objetivo principal es implementar y validar el modelo diseñado, demostrando que refleja de manera precisa la operación logística y que es capaz de predecir rutas y costos de forma confiable. Cada equipo deberá implementar el modelo utilizando herramientas como *Python* (`Pyomo` o `PuLP`) o *GAMS*, resolviendo casos de prueba que simulan diferentes escenarios operacionales. A través de esta implementación, se espera obtener resultados que permitan evaluar la viabilidad y precisión del modelo, así como identificar oportunidades de mejora.

La validación de esta etapa es clave, ya que los equipos analizarán no solo los resultados de las rutas optimizadas, sino también los costos involucrados, identificando discrepancias entre el modelo y las operaciones reales. Estos análisis permitirán a Seneca Libre tomar decisiones estratégicas informadas, fortaleciendo su posición en el mercado.

Además de validar el modelo, esta fase establece las bases para futuras simulaciones y ajustes, con el objetivo de reducir costos, mejorar tiempos de entrega y elevar los estándares de servicio, asegurando la sostenibilidad en el competitivo mercado del comercio electrónico.

2 Objetivos Específicos de la Etapa 2 & 3

- Implementar el modelo matemático utilizando herramientas de programación como *Python* (`Pyomo` o `PuLP`) o *GAMS*.
- Resolver y analizar varios casos de prueba que reflejen escenarios operacionales representativos, validando la efectividad del modelo en la optimización de costos.
- Explorar y abordar escenarios avanzados, incluyendo:
  - Centros de distribución con capacidades de oferta limitada.
  - Demanda diferenciada de múltiples tipos de productos.
  - Integración de **nodos de recarga** para vehículos eléctricos en la optimización.
  - Resolución del problema utilizando **algoritmos genéticos** como método heurístico para escenarios simplificados.
- Realizar un análisis detallado de los resultados obtenidos, enfocándose en:
  - Optimización de la flota y reducción de tiempos de entrega.
  - Identificación de patrones y oportunidades de mejora.
  - Sugerencias para ajustes y mejoras futuras en el modelo.

3 Datos y Recursos Disponibles

Para la implementación y resolución del modelo de ruteo vehicular en esta etapa, se cuenta con diversos datos de entrada que describen los puntos de entrega, centros de distribución, vehículos y nodos de recarga. Aunque algunos datos adicionales, como los tipos de productos y las capacidades específicas de los centros de distribución, no son críticos para los primeros casos de prueba, se incluyen a continuación para una referencia completa.

3.1 Datos de Puntos de Entrega

Cada cliente se representa mediante coordenadas geográficas, utilizando la latitud y longitud para determinar su ubicación en el mapa. El conjunto de datos incluye:

- **LocationID:** Un identificador único para cada nodo. (Puede ser un cliente o un centro de distribución o un nodo de recarga).
- **ClientID:** Un identificador único para cada cliente.

- **Demanda**[kg]**:** Cantidad de producto solicitada por el cliente, expresada en kilogramos (kg). (* **Nota:** Para uno de los casos donde hay multiples productos, se debe considerar la demanda de cada producto por separado **i.e** 'Producto-Type-A', 'Producto-Type-B', 'Producto-Type-C')

- **Longitud y Latitud:** Coordenadas geográficas en grados decimales, expresadas como $\text{Longitud}_{\text{cliente}}$ (°) y $\text{Latitud}_{\text{cliente}}$ (°).

3.2 Datos de Centros de Distribución

Los centros de distribución (o "depots") se representan con su ubicación geográfica y la capacidad de almacenamiento por tipo de producto:

- **LocationID:** Un identificador único para cada nodo. (Puede ser un cliente o un centro de distribución o un nodo de recarga).

- **DepotID:** Un identificador único para cada centro de distribución.

- **Longitud y Latitud:** Coordenadas geográficas en grados decimales, expresadas como $\text{Longitud}_{\text{depot}}$ (°) y $\text{Latitud}_{\text{depot}}$ (°).

- **Capacidad de Producto**[kg]**:** Capacidad de almacenamiento por cada tipo de producto (* **Nota** A, B y C Dependiendo si tenemos uno mas productos.) en cada centro de distribución, expresada en unidades de producto. Aunque se incluyen en el conjunto de datos, estas capacidades no son relevantes para los primeros casos de prueba.

3.3 Datos Nodos de Recarga

- **LocationID:** Un identificador único para cada nodo. (Puede ser un cliente o un centro de distribución o un nodo de recarga).

- **RechargeNodeID:** Un identificador único para cada centro de distribución.

- **Longitud y Latitud:** Coordenadas geográficas en grados decimales, expresadas como $\text{Longitud}_{\text{depot}}$ (°) y $\text{Latitud}_{\text{depot}}$ (°).

3.4 Datos de Vehículos

Los datos de vehículos disponibles incluyen información detallada sobre el tipo de vehículo, capacidad de carga, rango operativo y costos asociados. A continuación, se describen los parámetros específicos de cada tipo de vehículo:

- **VehicleType:** Tipo de vehículo. Se incluyen tres tipos:
  - *Gas Car:* Vehículo de gasolina con parámetros específicos de eficiencia y costos.
  - *Drone:* Vehículo aéreo no tripulado que utiliza electricidad como fuente de energía.
  - *Solar EV:* Vehículo eléctrico con capacidad de recarga solar.

- **Capacidad** ($C_{\text{vehículo}}$): Capacidad máxima de carga de cada vehículo, expresada en kilogramos (kg). Cada vehículo tiene una capacidad específica para transportar carga.
- **Rango** ($R_{\text{vehículo}}$): Distancia máxima que puede recorrer el vehículo con una carga completa de combustible o energía, expresada en kilómetros (km).
- **Tarifa de Flete** (Freight Rate): Costo por kilómetro recorrido por cada tipo de vehículo, expresado en pesos colombianos por kilómetro (COP/km).
- **Tarifa de Tiempo** (Time Rate): Costo por minuto de operación del vehículo, expresado en pesos colombianos por minuto (COP/min).
- **Costo de Mantenimiento Diario** (Daily Maintenance): Costo diario de mantenimiento del vehículo, expresado en pesos colombianos por día (COP/día).
- **Costo de Recarga/Combustible** (Recharge/Fuel Cost): Costo de recarga o combustible por unidad (galón o kWh), expresado en pesos colombianos (COP/gal o COP/kWh).
- **Tiempo de Recarga/Combustible** (Recharge/Fuel Time): Tiempo necesario para recargar un 10% de la capacidad total de energía o combustible del vehículo, expresado en minutos (min/10% de carga).
- **Velocidad Promedio** (Avg. Speed): Velocidad promedio del vehículo, expresada en kilómetros por hora (km/h). (* **Nota:** Este solo esta incluido para los drones ya que para los vehiculos terrestres la velocidad promedio no es la más importante.)
- **Eficiencia de Combustible** (Gas Efficiency): Eficiencia del vehículo en cuanto a consumo de gasolina, expresada en kilómetros por galón (km/gal) para vehículos de gasolina.
- **Eficiencia Energética** (Electricity Efficiency): Eficiencia en consumo de electricidad, expresada en kilovatios hora por kilómetro (kWh/km) para vehículos eléctricos y drones.

Formula para hallar el combustible o energia que se necesita para recorrer una distancia d en un vehiculo:

$$\text{Combustible/Energía} = \frac{d}{\text{Eficiencia de Combustible/Energía}} \tag{1}$$

Formula para hallar el tiempo que se necesita para recargar el combustible o energia que se necesita para recorrer una distancia d en un vehiculo:

$$\text{Capacidad Total de Energía o Combustible} = \frac{\text{Rango del Vehículo}}{\text{Eficiencia del Vehículo}} \tag{2}$$

$$\text{Energía o Combustible al 10\%} = 0.1 \times \text{Capacidad Total de Energía o Combustible} \tag{3}$$

$$\text{Tiempo de Recarga o Reabastecimiento} = \frac{\text{Energía o Combustible al 10\%}}{\text{Tasa de Recarga o Reabastecimiento}} \tag{4}$$

Los datos específicos para cada tipo de vehículo se detallan en la siguiente tabla:

Parámetro	Gas Car	Drone	Solar EV
Freight Rate [COP/km]	5000	500	4000
Time Rate [COP/min]	500	500	500
Daily Maintenance [COP/day]	30000	3000	21000
Recharge/Fuel Cost [COP/(gal or kWh)]	16000	220.73	-
Recharge/Fuel Time [min/10% charge]	0.1	2	-
Avg. Speed [km/h]	-	40	-
Gas Efficiency [km/gal]	10	-	-
Electricity Efficiency [kWh/km]	-	0.15	0.15

Table 1: Parámetros de Costos y Eficiencia para Vehículos

3.4.1 Velocidad de Carga

Además de los costos de operación y recarga, el proceso de carga de cada vehículo tiene un costo asociado. Cada minuto de carga tiene un costo de 500 COP, y se puede cargar un promedio de 5 kg de carga por minuto, en función del trabajo de un empleado encargado de la carga de los vehículos.

Definimos el costo total de carga (C_{carga}) para un vehículo de la siguiente forma:

$$C_{\text{carga}} = \text{Carga} \times \frac{500\,\text{COP}}{\text{kg/min}} \tag{5}$$

donde *Carga* es la cantidad de peso en kilogramos (kg) que se carga en el vehículo en un tiempo determinado.

3.5 Datos de Nodos de Recarga

Los nodos de recarga representan estaciones donde los vehículos pueden recargar combustible o energía. Estos nodos están definidos por:

- **NodeID:** Un identificador único para cada nodo. (Puede ser un cliente o un centro de distribución).
- **NodeID:** Un identificador único para cada nodo de recarga.
- **Longitud y Latitud:** Coordenadas geográficas en grados decimales, expresadas como $\text{Longitud}_{\text{nodo}}$ (°) y $\text{Latitud}_{\text{nodo}}$ (°).

3.6 Datos de Costos

Para el modelo de ruteo vehicular, los costos se componen principalmente de los gastos asociados a las distancias y tiempos operativos de cada vehículo. Aunque algunos datos de costos específicos están directamente relacionados con el tipo de vehículo, como las tarifas de flete y tiempo, otros deben calcularse en función de la distancia recorrida y el tiempo empleado en cada ruta.

El costo operativo de cada vehículo en una ruta particular se puede expresar en función de la distancia (d, en km) y el costo por kilómetro (c_{km}), obteniendo el costo total (C_{total}) mediante la siguiente ecuación:

$$C_{\text{total}} = c_{\text{km}} \times d \tag{6}$$

donde c_{km} es el costo por kilómetro de operación para el tipo específico de vehículo utilizado, y d representa la distancia total recorrida en kilómetros.

De igual manera, se debe considerar el tiempo de operación (t, en minutos) y el costo por minuto (c_{min}) asociado a cada vehículo para obtener el costo total en función del tiempo:

$$C_{\text{tiempo}} = c_{\text{min}} \times t \tag{7}$$

donde c_{min} es el costo por minuto para el vehículo y t es el tiempo total de la ruta en minutos.

Por lo tanto, el costo total de operación ($C_{\text{operativo}}$) para una ruta puede calcularse como la suma de los costos de distancia y tiempo:

$$C_{\text{operativo}} = C_{\text{total}} + C_{\text{tiempo}} \tag{8}$$

Estos valores deberán ser calculados según las rutas específicas de cada caso de prueba, ya que las distancias y tiempos varían en cada escenario. Los demás costos, tales como mantenimiento diario, tarifas de recarga y costos asociados al proceso de carga, ya están cubiertos en los datos específicos del vehículo y no requieren cálculos adicionales en esta fase del análisis.

3.7 Herramientas de Ayuda, Recursos Adicionales y Consideraciones

Para el desarrollo del proyecto, se recomienda utilizar herramientas avanzadas de optimización que permitan resolver el modelo de ruteo vehicular de manera eficiente, especialmente en los casos de mayor complejidad y escalabilidad. A continuación, se presentan algunos de los solvers más utilizados y recursos adicionales para su instalación y configuración con Python y Pyomo.

3.7.1 Herramientas de Ayuda

A continuación se presentan algunas herramientas clave para resolver el modelo de ruteo vehicular de manera eficiente, especialmente en los casos de mayor complejidad y escalabilidad. Estas incluyen solvers de optimización de gran escala y recursos para la generación de datos sintéticos que ayudarán en el desarrollo y análisis del proyecto.

Solvers de Optimización a Gran Escala:

- **Gurobi:** Gurobi es uno de los solvers comerciales más potentes y ampliamente utilizados en optimización a gran escala. Soporta modelos lineales, no lineales y enteros mixtos. Es reconocido por su alta eficiencia y velocidad en problemas complejos. Para aprender a integrarlo con marcos de modelado de código abierto como `Pyomo`, consulte el siguiente tutorial: `https://support.gurobi.com/hc/en-us/articles/16316467014801-How-do-I-use-Gurobi-with-open-source-modeling-fram` Puede encontrar más información y recursos adicionales en su sitio web oficial: `https://www.gurobi.com/`.

- **CPLEX:** CPLEX, desarrollado por IBM, es otro solver de alto rendimiento ampliamente utilizado en optimización a gran escala. Ofrece soporte para una variedad de problemas complejos, incluyendo modelos lineales, no lineales y enteros mixtos. Para instalar y utilizar CPLEX con `Pyomo`, consulte el siguiente recurso: `https:`

//eseslab.com/Downloads/Installing-PYOMO-and-CPLEX.pdf. Puede acceder a la página oficial de CPLEX para información adicional: https://www.ibm.com/analytics/cplex-optimizer.

- **HiGHS:** HiGHS es un solver de código abierto y eficiente para resolver problemas de optimización lineal y entera mixta. En este caso, es el solver preferido debido a su ligereza y accesibilidad. Sin embargo, presenta algunas limitaciones, como la cantidad de nodos que puede evaluar simultáneamente. Existen tres métodos principales para instalarlo e integrarlo:

  1. **Mediante el ejecutable:** Aunque esta es una opción viable, actualmente el ejecutable de HiGHS no es reconocido directamente por Pyomo, lo que requiere realizar configuraciones adicionales para que funcione correctamente.
  2. **Con appsi:** Esta integración directa desde Pyomo es la más recomendada para usuarios de **Mac** y **Linux**, ya que simplifica considerablemente el proceso de configuración.
  3. **Con amplpy:** Para usuarios de **Windows**, esta opción resulta la más sencilla, ya que amplpy permite una integración rápida y eficiente con HiGHS.

  En el repositorio que se les proporcionará, encontrarán un tutorial detallado que explica cómo probar cada una de estas rutas para integrar HiGHS en sus proyectos. Es importante tener instalado un solver más avanzado, como Gurobi o CPLEX, para manejar problemas de mayor complejidad o con un número significativo de nodos, ya que HiGHS puede ser limitado en estas circunstancias. Recursos adicionales para integrar HiGHS con Python y Pyomo están disponibles en: https://ergo-code.github.io/HiGHS/dev/interfaces/python/, y más detalles técnicos pueden encontrarse en su página oficial: https://highs.dev/.

Opciones a habilitar en Pyomo para HiGHS: El solver **HiGHS** es una herramienta eficiente y de código abierto diseñada para resolver problemas de optimización lineal y entera mixta. A continuación, se describen las principales opciones que pueden habilitarse en Pyomo para ajustar su comportamiento:

- –**time-limit**: Define el límite máximo de tiempo para la resolución del problema, expresado en segundos. Ejemplo: solver.options['time$_l$imit'] = 300($estableceunlímitede5min$

- –**presolve**: Controla el nivel de preprocesamiento para simplificar el modelo antes de resolverlo. Puede ajustarse en niveles como 'off', 'on', o 'advanced'. Ejemplo: solver.options['presolve'] = 'on'.

- –**mip-rel-gap**: Define la tolerancia relativa para problemas de programación entera mixta (MIP). Ejemplo: solver.options['mip$_r$el$_g$ap'] = 0.01($toleraunasoluciónd entrodel1\%delóptimo$

Listing 1: Ejemplo de configuración para HiGHS en Pyomo

```
def solve_model(self):
    solver_name = "appsi_highs"
    solver = pyo.SolverFactory(solver_name)
    solver.options['parallel'] = 'on'
    solver.options['time_limit'] = 3600  # 1-hour time limit
```

```
solver.options['presolve'] = 'on'
solver.options['mip_rel_gap'] = 0.01  # 1% relative gap
solver.options['simplex_strategy'] = 1  # Dual simplex
solver.options['simplex_max_concurrency'] = 8  # Max
    concurrency
solver.options['mip_min_logging_interval'] = 10  # Log every
    10 seconds
solver.options['mip_heuristic_effort'] = 0.2  # Increase
    heuristic effort

result = solver.solve(self.model, tee=True)

# Check solver status
if result.solver.termination_condition == pyo.
    TerminationCondition.optimal:
    print("Optimal solution found.")
elif result.solver.termination_condition == pyo.
    TerminationCondition.maxTimeLimit:
    print("Time limit reached, solution may be suboptimal.")
else:
    print(f"Solver terminated with condition: {result.solver
        .termination_condition}")

print(result)
```

Las opciones presentadas en el ejemplo proporcionan configuraciones avanzadas para maximizar la eficiencia del solver y adaptarlo a diferentes escenarios operacionales. La verificación del estado del solver garantiza que los resultados sean interpretados correctamente.

Generador de Datos Sintéticos: Para ayudar en la creación de datos representativos y personalizados según los requisitos del problema, se ha proporcionado un generador de datos sintéticos. Este recurso permite generar datos similares a los de los casos de prueba, facilitando la experimentación con diferentes configuraciones sin necesidad de recolectar datos reales.

El generador de datos y los casos de prueba, tanto **estándares** como **especiales**, estarán ubicados en un repositorio de **GitHub**, el cual se compartirá con todos los estudiantes. Este repositorio centralizará las herramientas y recursos necesarios para el desarrollo del proyecto.

Además, en este repositorio se incluirá un *notebook* con todos los costos y parámetros relevantes de los vehículos, centros de distribución y clientes. Este archivo servirá como referencia clave para integrar el generador de datos con los modelos de optimización.

Estas herramientas están diseñadas para optimizar el proceso de modelado y análisis, permitiendo a los estudiantes enfocarse en el desarrollo y prueba de su modelo de ruteo vehicular de manera eficiente y efectiva.

3.7.2 Recursos Adicionales

Open Source Routing Machine (OSRM): OSRM es un motor de enrutamiento de código abierto que ofrece una API potente para calcular rutas óptimas, tiempos de viaje y

distancias entre puntos. Este recurso es fundamental para generar las matrices de costos para vehículos terrestres, ya que permite incorporar de manera realista las restricciones de infraestructura vial y las características de las rutas, como distancias reales y tiempos de viaje, que no pueden ser modelados con precisión usando simples cálculos de distancia euclidiana.

Para más información y acceso a la documentación oficial de OSRM, visite su sitio web: `http://project-osrm.org/`. Adicionalmente, puede aprender a utilizar OSRM y generar visualizaciones de rutas mediante el siguiente tutorial: `https://medium.com/walmartglobaltech/finding-and-plotting-optimal-route-using-open-source-api-in-python-` Este tutorial explica cómo obtener la distancia, la duración del viaje y la nube de puntos para representar gráficamente la ruta.

Por qué utilizar OSRM para Vehículos Terrestres A diferencia de los drones, los vehículos terrestres están sujetos a las restricciones de la infraestructura vial, como calles, intersecciones y límites de velocidad. Usar una API como OSRM permite calcular distancias y tiempos de viaje basados en la red vial real, en lugar de aproximaciones inexactas como la distancia euclidiana o haversiana. Esto es crucial para obtener una matriz de costos representativa que permita optimizar rutas de manera más efectiva y realista.

Uso de OSRM para la Generación de Matrices de Costos La API de OSRM debe ser utilizada para calcular las distancias y tiempos de viaje entre los nodos de clientes y los centros de distribución. Con esta información se generará la matriz de costos, tanto para vehículos de gasolina como eléctricos. Esta matriz será clave para la optimización de rutas basada en los costos operativos específicos de cada tipo de vehículo.

El tutorial compartido muestra cómo obtener, además de la distancia y la duración del viaje, la nube de puntos (*polyline*) que permite graficar las rutas en un mapa. Sin embargo, para propósitos de optimización y cálculo de la matriz de costos, no es necesario generar la nube de puntos en esta etapa. La idea es primero resolver el problema de optimización y, posteriormente, utilizar OSRM nuevamente para calcular y almacenar únicamente la nube de puntos para las rutas resultantes del problema resuelto. Este enfoque mejora la eficiencia al evitar el almacenamiento y procesamiento innecesario de datos intermedios.

Cálculo de la Matriz de Costos para Drones En el caso de los drones, no se ven afectados por las restricciones de infraestructura vial, por lo que se recomienda utilizar la distancia haversiana, una medida precisa de distancia entre dos puntos en la superficie terrestre, junto con la velocidad promedio del dron. Esto permite estimar de manera eficiente los costos de viaje sin depender de APIs externas, lo que simplifica la generación de la matriz de costos para estos vehículos aéreos.

3.8 Consideraciones Adicionales

Para garantizar que todos los estudiantes tengan la oportunidad de desarrollar e implementar su modelo de ruteo vehicular de manera efectiva, se brinda flexibilidad para simplificar o modificar ciertos aspectos del problema. Estas modificaciones deben estar justificadas y mantener una relación realista entre los datos y porcentajes relevantes, asegurando la coherencia y la posibilidad de comparar las implementaciones con las de sus compañeros.

A continuación, se presentan las consideraciones clave que deben tener en cuenta:

- **Simplificación de Múltiples Vehículos:** Los estudiantes pueden optar por simplificar el problema de múltiples vehículos a uno con un único tipo de vehículo, siempre y cuando la relación entre capacidades, costos y restricciones operativas se mantenga realista. La coherencia en los datos proporcionados debe justificarse y documentarse en el análisis del modelo.
- **Adaptación de Múltiples Centros de Distribución (Depots):** Es posible modelar el problema desde una perspectiva de un solo centro de distribución (*single depot*) o múltiples centros de distribución (*multi-depot*). Los datos proporcionados incluirán las ubicaciones sugeridas para los centros en ambos escenarios, permitiendo que cada grupo elija y justifique su enfoque.
- **Ajustes en los Costos y Parámetros Operativos:** Los costos y otros parámetros, como tiempos y capacidades, pueden ajustarse según las necesidades del modelo, siempre que las modificaciones se expliquen adecuadamente. Esto incluye mantener proporciones razonables entre los diferentes costos operativos, asegurando que el modelo refleje decisiones viables y comparables.
- **Demanda y Ubicación de los Clientes (Inamovible):** Lo único que debe mantenerse fijo en todos los modelos es la **demanda** de los clientes y la **ubicación geográfica** de cada uno. Estos elementos garantizan que las soluciones sean comparables y que los resultados reflejen el mismo problema base.
- **Experimentación y Justificación:** Los estudiantes son libres de experimentar con simplificaciones o modificaciones en su modelado, siempre que estas sean justificadas y documentadas. Las decisiones tomadas deben ser consistentes con los objetivos del proyecto y permitir un análisis riguroso de las soluciones propuestas.

Además, los datos proporcionados incluirán opciones para determinar las ubicaciones en un escenario *multi-depot* versus un escenario *single depot*, lo que permitirá a los estudiantes decidir el enfoque más adecuado para su modelo. Estas recomendaciones fomentan la creatividad y flexibilidad en el diseño del modelo, garantizando que las soluciones puedan evaluarse de manera equitativa y que reflejen un balance entre realismo y viabilidad técnica.

3.8.1 Uso de Datos de Nodos de Recarga

En los casos donde los vehículos necesiten recargar combustible o energía durante sus rutas, los estudiantes deben incluir los nodos de recarga en el modelo. Los nodos de recarga están identificados en los datos proporcionados y cuentan con coordenadas geográficas. Estos nodos deben ser integrados como puntos intermedios en las rutas, asegurando que los vehículos puedan recargar según sus necesidades de combustible o electricidad, especialmente en escenarios de larga distancia o para vehículos con un rango limitado, como los drones y los vehículos eléctricos.

3.8.2 Ajustes en la Matriz de Costos y Restricciones de Distancia

Para el cálculo de la matriz de costos, es importante diferenciar entre los vehículos terrestres y los drones:

- **Vehículos Terrestres (Gas Car, Solar EV):** Se recomienda utilizar la API de OSRM para calcular las distancias y tiempos de viaje entre los nodos de clientes y los centros de distribución. Esto generará una matriz de costos precisa basada en las distancias reales de las rutas vehiculares.

- **Drones:** Para los drones, que no siguen rutas de carretera, se sugiere calcular la matriz de costos utilizando la distancia haversiana, basada en las coordenadas geográficas de los nodos y la velocidad promedio del dron. Esto asegura que los costos sean representativos de la realidad operativa de los drones.

3.8.3 Coherencia y Representatividad de los Datos

Es crucial que cualquier ajuste realizado en el modelo respete la coherencia y la representatividad de los datos. Los cambios deben alinearse con las restricciones físicas y ser coherentes con los parámetros definidos para cada tipo de vehículo y depot. Esto incluye:

- La capacidad de carga y el rango operativo de los vehículos deben estar bien definidos y aplicarse adecuadamente en cada caso de prueba.

- Las limitaciones de suministro en cada depot deben respetarse, asegurando que cada uno no exceda su capacidad máxima para cada tipo de producto.

Estas consideraciones adicionales ayudarán a asegurar que el modelo se adapte de manera realista y efectiva a las condiciones operativas de Seneca Libre, reflejando las restricciones logísticas y de recursos en cada escenario del proyecto. El unico valor que no es modificable son los clientes, y en dado caso que su modelado actual no sea suficiente para llegar a una solución de los casos estandares o especiales tendran que modificar el modelado hasta que lleguen a una solución.

4 Descripción de los Escenarios de Prueba

Escenario	Número de Clientes	Número de Vehículos	Capacidad Promedio de Vehículos	Rango	Demanda por Cliente	Relación Vehículo-Cliente	Restricciones Adicionales
1. Escenario Base	24	12	Gas Car: 120 kg, EV: 96 kg, Drone: 25 kg	Gasolina: 150 km, EV: Infinito, Drone: 15 km	11 - 25 kg	2:1	Ninguna
2. Evaluación por Costos	30	6	Gas Car: 120 kg, EV: 96 kg, Drone: 25 kg	Gasolina: 150 km, EV: Infinito, Drone: 15 km	5 - 13 kg	5:1	Penalización según costo: 10 puntos adicionales = -0.1 en nota
3. Gestión de Oferta	9	6	Gas Car: 120 kg, EV: 96 kg, Drone: 25 kg	Gasolina: 150 km, EV: Infinito, Drone: 15 km	5 - 13 kg	1.5:1	Overflow en centros de distribución hasta 50%
4. Manejo de Productos	9	6	Grande	Gasolina: 150 km, EV: Infinito, Drone: 15 km	Variada para 3 tipos de productos	1.5:1	Vehículos y centros de distribución adaptados a la demanda mixta

Table 2: Resumen de los escenarios de prueba con sus características principales.

4.1 Escenario Base: Validación de Solución Factible

Este caso asegura que el modelo es capaz de producir rutas válidas y respetar las restricciones fundamentales:

- Sin subrutas y respetando las capacidades de los vehículos.
- Configuración inicial con 24 clientes, 12 vehículos (Gasolina, EV y Drones) y demandas entre 11 kg y 25 kg.
- Capacidad y rangos predefinidos según el tipo de vehículo.

Objetivo: Garantizar que el modelo funcione correctamente antes de avanzar hacia escenarios más complejos.

4.2 Escenario 2: Evaluación por Costos

Se centra en optimizar los costos totales del modelo:

- 30 clientes y 6 vehículos, con una relación cliente-vehículo de 5:1.
- Penalización basada en eficiencia de costos para fomentar la optimización.
- Demanda de clientes entre 5 y 13 kg.

Objetivo: Identificar soluciones más económicas y premiar la eficiencia en el diseño de rutas.

4.3 Escenario 3: Gestión de Oferta en Centros de Distribución

Introduce restricciones adicionales en los centros de distribución:

- Solo 9 clientes con demandas específicas.
- Restricción adicional de overflow en centros hasta un 50%.
- Relación cliente-vehículo ajustada a 1.5:1.

Objetivo: Evaluar la capacidad del modelo para manejar limitaciones de oferta sin comprometer la factibilidad de las soluciones.

4.4 Escenario 4: Manejo de Múltiples Productos

Introduce la complejidad de manejar distintos tipos de productos:

- 9 clientes, 6 vehículos.
- Se consideran tres tipos de productos y capacidades aumentadas en vehículos y centros de distribución.
- Restricciones mixtas relacionadas con los productos.

Objetivo: Analizar la habilidad del modelo para manejar múltiples demandas y restricciones simultáneamente.

4.5 Casos Especiales

Los casos especiales premian la creatividad y la aplicación de técnicas avanzadas. Permiten alcanzar calificaciones mayores a 5.0 si se resuelven correctamente.

4.5.1 Caso Especial 1: Nodos de Recarga

- Nodos de recarga introducidos estratégicamente para vehículos eléctricos.
- Tiempo y costo de recarga deben optimizarse.
- Configuración flexible para explorar rutas de largo alcance.

Objetivo: Desafiar a los estudiantes a extender la funcionalidad del modelo.

4.5.2 Caso Especial 2: Algoritmos Genéticos

- Implementación de heurísticas avanzadas para resolver una versión simplificada del problema.
- Comparación entre métodos exactos y heurísticos en tiempo y calidad de solución.

Objetivo: Fomentar la innovación en el diseño de soluciones y el uso de algoritmos heurísticos.

5 Entregables

Para esta etapa del proyecto, se requiere una serie de entregables que cubren desde la implementación del modelo hasta el análisis detallado de los resultados obtenidos en cada caso de prueba. El objetivo es asegurar que los estudiantes desarrollen una solución robusta y adaptable, capaz de manejar diferentes escenarios sin necesidad de ajustar el modelo para cada caso específico. A continuación, se detallan los entregables requeridos:

5.1 Implementación del Modelo

Se espera una versión actualizada del modelo matemático entregado en la fase anterior. Esta actualización debe incorporar los ajustes necesarios para soportar todos los casos de prueba de manera flexible. Los estudiantes deben documentar:

- Los cambios realizados al modelo para adaptarse a los distintos escenarios.
- La justificación de cada cambio y las ventajas identificadas al implementar estos ajustes.
- Asegurarse de que el modelo sea generalizable y no requiera modificaciones adicionales para cada caso de prueba.

El modelo debe implementarse en una clase en el lenguaje de programación elegido, con el código organizado para recibir los mismos formatos de archivos que los datos de los casos de prueba, facilitando así su evaluación. Cualquier intervención manual en los datos podría resultar en penalizaciones.

5.2 Rutas de los Vehículos

Para cada caso, los estudiantes deben entregar un archivo en formato `.csv` que documente las rutas seguidas por cada vehículo de manera clara y estructurada. Este archivo permitirá evaluar las secuencias de nodos visitados por cada vehículo y analizar la efectividad del modelo en el cumplimiento de las demandas logísticas.

Formato del Nombre del Archivo El nombre del archivo debe seguir el siguiente formato:

`<nombre-del-grupo>-caso-<tipo-de-caso>-<numero-del-caso>-ruta.csv`

donde:

- `<nombre-del-grupo>` es el nombre o identificador del equipo de estudiantes.
- `<tipo-de-caso>` indica si el caso es estándar o especial (`estandar` o `especial`).
- `<numero-del-caso>` es el número del caso correspondiente (por ejemplo, `1`, `2`, etc.).

Por ejemplo, un archivo de rutas para el primer caso estándar del equipo "Optimizadores" sería nombrado como:

`Optimizadores-caso-estandar-1-ruta.csv`

Mientras que un archivo para el primer caso especial sería:

`Optimizadores-caso-especial-1-ruta.csv`

Formato del Contenido del Archivo Cada archivo debe seguir el siguiente formato de columnas en el interior, organizado en formato `csv` para facilitar su lectura y análisis:

`ID-Vehiculo, ID-Origen, ID-Destino`

donde:

- `ID-Vehiculo` es el identificador único del vehículo que realiza la ruta.
- `ID-Origen` es el identificador del nodo de inicio de la ruta o de la parada actual.
- `ID-Destino` es el identificador del siguiente nodo al que se dirige el vehículo en su ruta.

Cada línea del archivo representa una transición de un nodo a otro para un vehículo específico, registrando toda la secuencia de paradas hasta la finalización de su ruta. Este formato permite analizar el trayecto detallado de cada vehículo de manera sencilla.

Ejemplo de Contenido A continuación, se muestra un ejemplo del contenido esperado dentro de un archivo de rutas para ilustrar el formato:

```
ID-Vehiculo,ID-Origen,ID-Destino
V01,N001,N002
V01,N002,N005
V01,N005,N003
V02,N003,N004
V02,N004,N006
V02,N006,N001
```

En este ejemplo:

- El vehículo `V01` comienza su ruta en el nodo `N001`, se desplaza al nodo `N002`, y así sucesivamente hasta completar su ruta.
- El vehículo `V02` sigue su propia secuencia de nodos comenzando en `N003` y terminando en `N001`.

5.3 Valor de la Función Objetivo

Para cada caso de prueba, se debe entregar un archivo con el valor de la función objetivo obtenida, que debe incluirse en un archivo con el prefijo `Caso_Objetivo` o `CasoEspecial_Objetivo` (por ejemplo, `Caso1_Objetivo.txt`). Este archivo debe contener:

- El valor total de la función objetivo para el caso.

5.4 Reporte Costos Operacionales

Se debe incluir un reporte detallado de los costos operacionales que contribuyen al valor de la función objetivo. Este informe debe desglosar cada tipo de costo involucrado, tales como costos de distancia, tiempo de operación, mantenimiento diario, recarga o combustible, y cualquier otro costo relevante en el cálculo de la función objetivo.

5.5 Visualización de Rutas

Para cada caso, se debe generar una visualización gráfica que muestre las rutas seguidas por cada vehículo. La visualización debe permitir identificar el recorrido de cada vehículo, mostrando claramente los nodos de inicio y los destinos. Los detalles sobre los requisitos específicos de la visualización se amplían en la sección de calificación y bonos.

5.6 Análisis de Resultados

Se debe presentar un análisis exhaustivo de los resultados obtenidos en cada caso de prueba, incluyendo:

- Identificación de patrones o tendencias observadas en las rutas optimizadas.
- Problemas o desafíos encontrados en la modelación.
- Mejoras potenciales para optimizar los resultados y sugerencias para ajustes futuros.

5.7 Código del Modelo

El código del modelo debe organizarse en una clase, la cual debe estar diseñada para recibir los datos en el formato especificado para cada caso de prueba. Esto facilitará su uso y revisión, permitiendo a los evaluadores ejecutar el modelo sin modificaciones adicionales. El modelo debe cumplir con los siguientes requisitos:

- El modelo debe ser capaz de adaptarse a cualquier caso de prueba utilizando el mismo código base.
- Los datos deben ser procesados directamente sin intervención manual para asegurar la reproducibilidad.
- La clase debe estar bien documentada, con instrucciones claras sobre su ejecución y los parámetros de entrada necesarios.

6 Sistema de Calificación

Este sistema de calificación se ajusta para reflejar un enfoque equilibrado y comprensivo de las dos entregas del proyecto, garantizando que se evalúen tanto los resultados prácticos como la calidad técnica del trabajo. El sistema fomenta la creatividad, innovación y excelencia técnica, con un peso significativo para los casos estándar y la calidad del modelado matemático.

6.1 Estructura de la Calificación

El proyecto se divide en dos entregas principales, cada una con un peso del **50% de la nota total**. Ambas entregas evalúan los casos estándar, los cambios en el modelo matemático y la calidad del código. Las ponderaciones son las siguientes:

Categoría	Entrega 2	Entrega 3	Total
Casos Estándar	80%	80%	80%
Modelado y Cambios en el Modelo	15%	15%	15%
Calidad del Código	5%	5%	5%

Table 3: Distribución general del sistema de calificación.

6.2 Distribución de los Casos Estándar

Cada entrega evalúa dos casos estándar, distribuidos de la siguiente manera:

- **Entrega 2:**
  - Caso 1 (Base): **40%**
  - Caso 2 (Evaluación por Costos): **40%**
- **Entrega 3:**
  - Caso 3 (Gestión de Oferta en Centros de Distribución): **40%**
  - Caso 4 (Manejo de Múltiples Productos): **40%**

El total de los casos estándar representa el **80% de la nota de cada entrega**.

6.3 Evaluación Detallada

1. Cambios en el Modelo Matemático (15%) Se evaluará la capacidad del modelo para adaptarse a todos los casos estándar sin necesidad de modificaciones adicionales. Los criterios incluyen:

- Adaptabilidad a los nuevos requisitos de cada caso.
- Claridad y justificación de los cambios implementados.
- Impacto en la eficiencia del modelo.
- Flexibilidad del código y calidad de la implementación.

2. Calidad del Código (5%) La organización, claridad y funcionalidad del código se evaluarán de forma integral. Los aspectos a considerar incluyen:

- Estructura modular y uso eficiente de clases.
- Facilidad para parametrizar y reproducir los resultados.
- Cumplimiento con los formatos de entrada y salida especificados.
- Ausencia de errores manuales en los datos.

3. Casos Estándar (80%) Cada caso estándar tendrá entregables específicos, ponderados de la siguiente manera:

- **Archivo CSV de Rutas (40%):** Validez y coherencia de las rutas generadas.
- **Valor de la Función Objetivo y Análisis de Costos (15%):** Documentación precisa de la función objetivo y desglose detallado de costos.
- **Visualización de Rutas (25%):** Claridad, diseño y creatividad en la representación gráfica.
- **Análisis de Resultados (20%):** Interpretación de patrones, problemas y posibles mejoras.

6.4 Incentivos y Bonificaciones

Se mantienen las bonificaciones y premios para reconocer trabajos excepcionales:

- **Bonificación de 0.5 puntos:** Para las visualizaciones más creativas y efectivas, que destaquen por su calidad visual y comunicativa.
- **Casos Especiales:** 20% del 70% la nota del proyecto.