Universidad Pontificia Bolivariana

Escuela de Ingeniería

Facultad de TIC

Optimización de Rutas de Ambulancias en Entornos Urbanos

Alexandra Vasco

Roy Sandoval

Técnicas de Optimización

2025

Abstract

Este proyecto desarrolla un modelo matemático de optimización basado en flujo multimercancía (multi-commodity flow) para determinar las rutas óptimas de ambulancias que responden a múltiples emergencias simultáneas en un área urbana. El modelo minimiza el costo total de operación del sistema, considerando los tiempos de viaje y los costos operativos diferenciados según el tipo de ambulancia (leve, media o crítica). La formulación respeta las capacidades de velocidad de las vías y garantiza que las velocidades requeridas para cada nivel de severidad sean factibles. La implementación se realizó en Python utilizando PuLP para la resolución del problema de programación lineal entera-mixta, integrando datos geográficos reales obtenidos mediante OSMnx. Los resultados se presentan mediante una aplicación interactiva desarrollada en Streamlit, que permite la configuración de parámetros y la visualización de rutas óptimas en un mapa.

Keywords: Optimización, Flujo Multimercancía, Enrutamiento de Vehículos, PuLP, OSMnx, Logística de Emergencias.

Contents

1 Introducción

Este proyecto desarrolla un modelo matemático de optimización basado en flujo multimercancía para determinar las rutas óptimas de ambulancias que responden a múltiples emergencias simultáneas en un área urbana de aproximadamente un kilómetro cuadrado. El modelo minimiza el costo total de operación del sistema considerando los tiempos de viaje y los costos operativos diferenciados según el tipo de ambulancia (leve, media o crítica), mientras respeta las capacidades de velocidad de las vías y garantiza tiempos de respuesta adecuados según la severidad de cada emergencia.

La implementación se realizó utilizando Python con la librería PuLP para la resolución del problema de programación lineal entera binaria, integrando datos geográficos reales obtenidos mediante OSMnx. El presente informe describe la formulación matemática del modelo, el procedimiento de implementación computacional, y los resultados obtenidos en diferentes escenarios de prueba, presentados mediante una aplicación interactiva desarrollada en Streamlit que permite visualizar las rutas óptimas en mapas.

2 Descripción del Área de Estudio y Datos Utilizados

El modelo se implementa sobre la red vial urbana de Medellín, Colombia, centrada en las coordenadas de la Clínica Medellín Occidente (6.2331°N, 75.5839°W). Se selecciona un área circular con radio configurable entre 400 y 800 metros, lo que permite analizar zonas compactas del entramado urbano donde convergen diferentes tipos de vías: avenidas principales, calles secundarias y vías residenciales.

Los datos de la red vial se obtienen de OpenStreetMap (OSM), una base de datos geográfica colaborativa de acceso libre. OSM proporciona información detallada sobre la topología de las calles, incluyendo ubicación de intersecciones, conectividad entre segmentos viales, longitudes de calles, y en algunos casos, características como tipo de vía y número de carriles.

Para acceder a estos datos se utiliza la librería OSMnx, que facilita la descarga, construcción y análisis de redes de calles desde OSM. OSMnx convierte automáticamente los datos geográficos en un grafo dirigido donde los nodos representan intersecciones y los arcos representan segmentos de calle con dirección específica.

Los datos obtenidos incluyen:

- **Nodos**: Intersecciones de calles con coordenadas geográficas (latitud, longitud).
- **Arcos dirigidos**: Segmentos de calle con información de:

- Longitud en metros.
- Nodo origen y nodo destino.
- Geometría de la vía (coordenadas intermedias).

3 Formulación Matemática del Modelo de Optimización

El problema se modela como un flujo multimercancía donde cada *commodity* representa una ambulancia con una ruta única desde la base de origen hasta su emergencia asignada.

3.1 Definición de Conjuntos, Parámetros y Variables

3.1.1 Conjuntos

- **N:** Conjunto de nodos de la red vial (intersecciones).
- **A:** Conjunto de arcos dirigidos, donde $A = \{(i, j) \mid i \in N, j \in N\}$. Cada arco representa un segmento de calle transitable en dirección de i hacia j.
- **K:** Conjunto de commodities (emergencias). Cada $k \in K$ es una tupla $k = (d_k, s_k)$, donde:
  - $d_k \in N$ es el nodo de destino de la emergencia k.
  - $s_k \in \{\text{Leve, Media, Crítica}\}$ es el tipo de severidad.

3.1.2 Parámetros

- l_{ij}**:** Longitud del arco $(i, j) \in A$ en kilómetros.
- c_{ij}**:** Capacidad del arco $(i, j) \in A$ en km/h (velocidad máxima permitida). Se asigna aleatoriamente en el rango $[C_{\min}, C_{\max}]$.
- o**:** Nodo de origen $o \in N$ (base de ambulancias).
- r_k**:** Velocidad requerida para el commodity k en km/h. Generada aleatoriamente según la severidad s_k:
  - **Crítica:** $r_k \in [0.8 \cdot R_{\max}, R_{\max}]$
  - **Media:** $r_k \in [R_{\min} + 0.4(R_{\max} - R_{\min}), 0.9 \cdot R_{\max}]$
  - **Leve:** $r_k \in [R_{\min}, R_{\min} + 0.6(R_{\max} - R_{\min})]$

- α_s**:** Costo operativo en \$/hora según el tipo de severidad $s \in \{\text{Leve, Media, Crítica}\}$.
  - $\alpha_{\text{Leve}} = \$100/\text{hora}$
  - $\alpha_{\text{Media}} = \$250/\text{hora}$
  - $\alpha_{\text{Crítica}} = \$500/\text{hora}$

3.1.3 Variables de Decisión

- x_{ijk}**:** Variable binaria.

$$x_{ijk} \in \{0, 1\} \quad \forall (i, j) \in A, \forall k \in K$$

donde:

- $x_{ijk} = 1$ si la ruta del commodity k incluye el arco (i, j).
- $x_{ijk} = 0$ en caso contrario.

3.2 Función Objetivo

Se busca minimizar el costo total del sistema. El costo de cada ruta se calcula como su costo operativo por hora ($\alpha_{s(k)}$) multiplicado por el tiempo total de viaje. El tiempo de viaje en cada arco (i, j) para el commodity k se calcula dinámicamente como $\frac{l_{ij}}{r_k}$.

$$\min Z = \sum_{k \in K} \sum_{(i,j) \in A} \left(\alpha_{s(k)} \cdot \frac{l_{ij}}{r_k} \right) \cdot x_{ijk}$$

La sumatoria interna calcula el costo total para la ambulancia k, sumando el costo de atravesar cada arco (i, j) de su ruta. La sumatoria externa agrega los costos de todas las ambulancias despachadas.

3.3 Restricciones

3.3.1 Conservación del Flujo

Para cada commodity k y cada nodo i, el flujo neto debe satisfacer la demanda b_{ik}.

$$\sum_{j:(i,j) \in A} x_{ijk} - \sum_{j:(j,i) \in A} x_{jik} = b_{ik} \quad \forall i \in N, \forall k \in K$$

Donde el balance b_{ik} se define como:

$$b_{ik} = \begin{cases} 1 & \text{si } i = o \text{ (nodo origen)} \\ -1 & \text{si } i = d_k \text{ (nodo destino de } k) \\ 0 & \text{en otro caso (nodo intermedio)} \end{cases}$$

Esta restricción garantiza que se genere una única ruta continua desde el origen o hasta el destino d_k para cada emergencia k.

3.3.2 Restricción de Velocidad (Capacidad)

Una ambulancia k solo puede usar un arco (i, j) si su velocidad requerida r_k es menor o igual que la capacidad (velocidad máxima) c_{ij} de dicho arco.

$$r_k \cdot x_{ijk} \leq c_{ij} \quad \forall (i, j) \in A, \forall k \in K$$

Si $x_{ijk} = 1$ (el arco se usa), la restricción se activa: $r_k \leq c_{ij}$. Si $x_{ijk} = 0$ (el arco no se usa), la restricción se vuelve $0 \leq c_{ij}$, lo cual es trivialmente cierto.

4 Procedimiento de Implementación Computacional

La implementación se desarrolló en Python utilizando una arquitectura modular que separa responsabilidades en tres componentes principales.

4.1 Módulo de Gestión de Red (network.py)

Esta clase es responsable de la extracción, procesamiento y gestión de la red vial.

- **Descarga de redes viales**: Utiliza OSMnx para obtener grafos dirigidos desde OpenStreetMap (circular o rectangular) filtrados por tipo de red `drive`.
- **Procesamiento topológico**: Extrae la componente fuertemente conexa más grande del grafo para garantizar la alcanzabilidad.
- **Asignación de capacidades**: Genera aleatoriamente velocidades máximas (c_{ij}) para cada arco en el rango $[C_{\min}, C_{\max}]$.
- **Selección de nodos**: Implementa la selección aleatoria de nodos origen y destino.

- **Sistema de caché**: Serializa redes descargadas (pickle) para evitar descargas repetidas.

4.2 Módulo de Visualización (map_display.py)

Esta clase genera visualizaciones interactivas de la red y las rutas óptimas utilizando Folium.

- **Renderizado de red base:** Dibuja todos los arcos de la red con opacidad reducida como contexto.
- **Visualización de rutas optimizadas:** Superpone las rutas calculadas con colores diferenciados por severidad (Rojo: Crítica, Naranja: Media, Azul: Leve).
- **Marcadores geográficos:** Añade marcadores para el origen (base) y los destinos (emergencias).
- **Integración con Streamlit:** Retorna objetos Folium compatibles con el componente `st_folium`.

4.3 Módulo de Optimización

Este módulo implementa el modelo matemático y su resolución.

4.3.1 OptimizationData (data_interface.py)

Actúa como estructura de datos intermedia entre la red (OSMnx) y el modelo (PuLP).

- **Extracción de datos del grafo**: Método `from_network()` que extrae listas de nodos, arcos y atributos (longitud, capacidad).
- **Estructuración de emergencias**: Almacena las tuplas (d_k, s_k) para cada emergencia.

4.3.2 AmbulanceRoutingModel (model.py)

Clase central que implementa el modelo de la Sección 3.

- **Inicialización**: Recibe un objeto `OptimizationData` y crea un commodity k único por emergencia.
- **Configuración de parámetros**: Método `set_parameters()` que define α_s y genera las velocidades r_k para cada commodity.
- **Construcción del modelo PuLP (`build_model()`)**:

1. Crea el problema `LpProblem` con sentido `LpMinimize`.
2. Genera las variables binarias x_{ijk}.
3. Define la función objetivo (Sección 3.2).
4. Añade las restricciones de conservación de flujo (Sección 3.3.1).
5. Añade las restricciones de velocidad/capacidad (Sección 3.3.2).

- **Resolución del problema**: Implementa el método `solve()` de PuLP.
- **Extracción de resultados**: Métodos para procesar la solución óptima, identificar las variables $x_{ijk} = 1$, reconstruir las rutas (secuencias de nodos) y calcular métricas agregadas (distancia, tiempo, costo) por ruta.

4.4 Interfaz de Usuario (Streamlit)

La aplicación web se implementó utilizando Streamlit.

4.4.1 Estructura de la Interfaz

- **Panel de configuración lateral:** Contiene controles para todos los parámetros del modelo (geográficos, velocidades R y C, costos α, y cantidad de emergencias), además de los botones de recálculo.
- **Pestañas principales:**
  - **Map Visualization:** Carga y visualización del mapa con las rutas.
  - **Results:** Ejecución del modelo y presentación de métricas.
  - **About:** Documentación del proyecto.

4.4.2 Flujo de Ejecución

1. Usuario configura parámetros en el panel lateral.
2. Clic en "Load Network": `NetworkManager` descarga/carga la red, asigna capacidades c_{ij} y selecciona nodos.
3. `OptimizationData` estructura los datos.
4. En la pestaña "Results", clic en "Run Optimization": `AmbulanceRoutingModel` construye y resuelve el problema.

5. `MapVisualizer` dibuja las rutas óptimas en el mapa.
6. La interfaz despliega las métricas de la solución.

4.5 Tecnologías y Bibliotecas

- **Python 3.10+:** Lenguaje base.
- **PuLP:** Modelado de optimización.
- **OSMnx:** Extracción de redes viales.
- **NetworkX:** Manipulación de grafos.
- **Folium:** Visualización de mapas interactivos.
- **Streamlit:** Framework de la aplicación web.

5 Escenarios de Prueba y Resultados

Se diseñaron tres escenarios para validar el modelo: ideal, moderado y crítico.

5.1 Escenario 1: Escenario Ideal

Descripción: Baja demanda (K=2) y alta capacidad vial (tráfico fluido).

- Velocidades requeridas: $R_{\min} = 15$, $R_{\max} = 35$ km/h
- Capacidades viales: $C_{\min} = 50$, $C_{\max} = 90$ km/h
- Emergencias: $K = 2$

Resultados:

- **Estado:** Óptimo encontrado
- **Métricas globales:**
  - Costo total: **$13.70**
  - Distancia total: **2.39 km**
  - Tiempo total (agregado): **4.78 min**
  - Emergencias atendidas: **2/2 (100%)**

- **Detalles por emergencia:**
  - **Emergencia 1 (Leve):** Vel: 15.8 km/h · Dist: 1.24 km · T: 2.48 min · Costo: $4.14
  - **Emergencia 2 (Media):** Vel: 28.6 km/h · Dist: 1.15 km · T: 2.29 min · Costo: $9.56

5.2 Escenario 2: Escenario Moderado

Descripción: Demanda moderada (K=4) y capacidades viales medias.

- Velocidades requeridas: $R_{\min} = 18$, $R_{\max} = 40$ km/h
- Capacidades viales: $C_{\min} = 40$, $C_{\max} = 80$ km/h
- Emergencias: $K = 4$

Resultados:

- **Estado:** Óptimo encontrado
- **Métricas globales:**
  - Costo total: **$30.52**
  - Distancia total: **3.06 km**
  - Tiempo total (agregado): **6.12 min**
  - Emergencias atendidas: **4/4 (100%)**
- **Detalles por emergencia:**
  - **E1 (Leve):** 29.6 km/h · 0.91 km · 1.81 min · $3.02
  - **E2 (Media):** 32.3 km/h · 0.11 km · 0.21 min · $0.89
  - **E3 (Crítica):** 34.8 km/h · 1.15 km · 2.30 min · $19.13
  - **E4 (Media):** 33.2 km/h · 0.90 km · 1.79 min · $7.48

5.3 Escenario 3: Escenario Crítico

Descripción: Alta demanda (K=6) y capacidad vial limitada, simulando congestión.

- Velocidades requeridas: $R_{\min} = 25$, $R_{\max} = 50$ km/h
- Capacidades viales: $C_{\min} = 40$, $C_{\max} = 80$ km/h
- Emergencias: $K = 6$ (3 críticas, 2 medias, 1 leve)

Resultados:

- **Estado:** Óptimo encontrado
- **Métricas globales:**
  - Costo total: **$73.38**
  - Distancia total: **6.47 km**
  - Tiempo total (agregado): **12.94 min**
  - Emergencias atendidas: **6/6 (100%)**
- **Detalles por emergencia:**
  - **E1 (Crítica):** 49.5 km/h · 1.36 km · 2.72 min · $22.65
  - **E2 (Crítica):** 45.7 km/h · 0.47 km · 0.94 min · $7.80
  - **E3 (Media):** 38.9 km/h · 0.68 km · 1.36 min · $5.65
  - **E4 (Crítica):** 45.2 km/h · 1.31 km · 2.62 min · $21.86
  - **E5 (Media):** 41.7 km/h · 1.31 km · 2.62 min · $10.93
  - **E6 (Leve):** 28.6 km/h · 1.34 km · 2.69 min · $4.48

5.4 Resumen Comparativo

Table 1: Resumen Comparativo de Escenarios

Escenario	Emergencias (K)	Estado	Costo total ($)	Tiempo total (min)	Distancia tota
Ideal	2	Factible	13.70	4.78	2.39
Moderado	4	Factible	30.52	6.12	3.06
Crítico	6	Factible	73.38	12.94	6.47

6 Conclusiones

Este proyecto ha logrado diseñar, formular e implementar exitosamente un modelo de optimización de flujo multimercancía para el enrutamiento de ambulancias en respuesta a emergencias. La solución, construida con PuLP y datos viales de OSMnx, demuestra ser capaz de generar rutas óptimas que minimizan el costo operativo total, basándose en una función objetivo que pondera el costo por hora de cada tipo de ambulancia por su tiempo de viaje.

El análisis de los tres escenarios (Ideal, Moderado y Crítico) valida la robustez del modelo. Los resultados comparativos de la Tabla 1 son particularmente reveladores: el costo total no escala linealmente con el número de emergencias, sino que crece de forma más pronunciada a medida que la red se congestiona. Esto se debe a la interacción de las dos restricciones de velocidad: la alta velocidad requerida (r_k) para emergencias críticas, combinada con capacidades viales (c_{ij}) limitadas en escenarios congestionados, obliga al optimizador a encontrar rutas que son geométricamente más largas pero "factibles" en términos de velocidad. El modelo identifica correctamente rutas no obvias que un simple algoritmo de "camino más corto" (basado solo en distancia) ignoraría.

Es importante reconocer las limitaciones de la presente formulación. El modelo asume capacidades viales estáticas (c_{ij}), las cuales no reflejan la naturaleza dinámica de la congestión del tráfico, que varía según la hora del día. Asimismo, se modela un único punto de origen (base) y se asume una disponibilidad ilimitada de ambulancias de cada tipo, lo cual es una simplificación de un sistema real de despacho.

Como trabajo futuro, el modelo podría extenderse en varias direcciones clave: 1) La integración de datos de tráfico en tiempo real o históricos para implementar capacidades (c_{ij}) que dependan del tiempo. 2) La expansión a un modelo multi-origen para gestionar una flota de ambulancias estacionadas en múltiples hospitales o bases. 3) La adición de restricciones de capacidad en el nodo origen para limitar el número de ambulancias disponibles de cada tipo, transformando el problema en uno de asignación de recursos más complejo.

En conclusión, este trabajo constituye una base sólida para un sistema de respuesta a emergencias. La aplicación interactiva en Streamlit cierra exitosamente la brecha entre la teoría de optimización compleja y una herramienta de apoyo a la toma de decisiones, visual y práctica, para la logística urbana.