Proyecto A etapa 2: Optimizacion en logistica para LogistiCo

Marcos España, Jorge Bustamante, Jairo Fierro

Mayo 2025

1 Introduction

LogistiCo es una empresa líder de soluciones logísticas. El objetivo de este proyecto es minimizar los costos de operación y transporte al mismo tiempo que se garantiza un servicio eficiente dentro de las restricciones que hay en una ciudad como Bogotá.

2 Implementación en pyomo

NOTA: La implementación del código la hicimos en el archivo entrega2.ipynb donde para generar los archivos de verificación para cada caso y el mapa se sacan ejecutando el programa con los archivos distintos de cada caso.

2.1 Lectura de datos

Empezamos cargando los datos, en este ejemplo los datos del caso3.

A partir del archivo depots.csv, se construye el conjunto C con los identificadores de los centros de distribución y se crea un diccionario CAP que asocia a cada centro su capacidad máxima.

```
C=list(depots_df["DepotID"])
CAP=dict(zip(depots_df["DepotID"],depots_df["Capacity"]))
```

Figure 1: Conjunto centros de distribución.

A partir del archivo clients.csv, se define el conjunto K con los identificadores de los clientes y el diccionario DEM que asigna la demanda requerida por cada uno.

```
K=list(clients_df["ClientID"])
DEM=dict(zip(clients_df["ClientID"],clients_df["Demand"]))  # Esta es la demanda que solicita ese cliente
```

Figure 2: Conjunto Zonas de entrega.

Se definen el conunto V de los vehículos.

```
V=list(vehicles_df.index)
Q = dict(zip(V, vehicles_df["Capacity"]))


R = dict(zip(V, vehicles_df["Range"]))
```

Figure 3: Conjunto vehículos

2.2 Cálculo de distancias

Se calcula la distancia entre cada centro de distribución y cada cliente utilizando la fórmula de Haversine, la cual estima la distancia sobre la superficie terrestre.Posteriormente se recorre cada par de centro–cliente para calcular y almacenar las distancias en kilómetros.

```
def dist_haversiana(lon1,lat1,lon2,lat2):
    R=6371.0 #radio tierra
    lon1,lat1,lon2,lat2=map(np.radians,[lon1,lat1,lon2,lat2])
    dlon=lon2-lon1
    dlat=lat2-lat1
    a = np.sin(dlat / 2.0)**2 + np.cos(lat1) * np.cos(lat2) * np.sin(dlon / 2.0)**2
    c = 2 * np.arcsin(np.sqrt(a))
    return R * c
```

Figure 4: Distancia Haversine.

2.3 Definición del modelo, variables y restricciones

Se crea una instancia del modelo en concreto y luego se definen los conjuntos C, K, V y N en el modelo.

```
model=ConcreteModel()

#Definicion de conjuntos
model.C= Set(initialize=C) # Centros de distribucion
model.K = Set(initialize=K) # Clientes
model.V = Set(initialize=V) # Vehículos
model.N = model.C | model.K    # Todos los nodos CDs + Clientes el | hace la unicoon de los conjuntos
```

Figure 5: Modelo y conjuntos.

Ahora se inicializan los datos procesados anteriormente como parametros del modelo.

```
# Parámetros
model.CAP = Param(model.C, initialize=CAP)
model.DEM = Param(model.K, initialize=DEM)
model.Q = Param(model.V, initialize=Q)
model.R = Param(model.V, initialize=R)
```

Figure 6: Parametros.

```
# Distancias entre centros y clientes
model.DIST = Param(model.C, model.K, initialize=DIST, within=NonNegativeReals)
```

Figure 7: Distancias entre centros y clientes.

De acuerdo al análisis de costos se definen los costos unitarios del modelo(COP/km).

```
Pf = 123.12  # COP/km (combustible)
Ft = 823     # COP/km (flete)
Cm = 700     # COP/km (mantenimiento)
```

Figure 8: Precios.

2.3.1 Variables de decisión

Definimos las variables de decisión del problema.

```
model.x =Var(model.C,model.K,model.V,domain=Binary) #X{i,j,l}
model.y= Var(model.C,model.V,domain=Binary) #y{i,l}
model.I=Var(model.C,domain=NonNegativeReals) # Inventario asignado a los centros de distribucion
model.u= Var(model.K,model.V,domain=NonNegativeReals) # Carga entregada en cada punto de distribucion
```

Figure 9: Variables de decisión.

2.3.2 Función objetivo

Definimos la función objetivo a minimizar del problema.

```
def objetivo(model):
    return sum(
        (Pf+Ft+Cm)* model.DIST[i,j]*model.x[i,j,l]
        for i in model.C for j in model.K for l in model.V
    )

model.OBJ=Objective(rule=objetivo,sense=minimize)
```

Figure 10: Función objetivo

2.3.3 Restricciones

Luego de definir la función objetivo, definimos las restricciones.

Restricción encargada de que en el modelo se respeten los centros de distribución.

```
def restriccionCapacidad_CentroDistribucion(model,i):
  return model.I[i]<=model.CAP[i]

model.RestriccionCapacidad_CentroDistribucion=Constraint(model.C, rule=restriccionCapacidad_CentroDistribucion)
```

Figure 11: Restricción capacidad de los centros de distribución.

Restricción encargada de que se cumplan las demandas de los clientes.

```
def restriccion_demanda(model, j):
    return sum(model.u[j, l] for l in model.V) == model.DEM[j]

model.RestriccionDemanda = Constraint(model.K, rule=restriccion_demanda)
```

Figure 12: Restricción demanda de los cientes.

Restricción encargada de la capacidad del vehículo.

```
def restriccion_capacidad_vehiculo(model, l):
    return sum(model.u[j, l] for j in model.K) <= sum(model.Q[l] * model.y[i, l] for i in model.C)

model.RestriccionCapacidadVehiculo = Constraint(model.V, rule=restriccion_capacidad_vehiculo)
```

Figure 13: Restricción capacidad del vehículo.

Restricción para que cada vehículo sea asignado a un único centro de distribución.

```
def restriccion_asignacion_vehiculos(model, l):
    return sum(model.y[i, l] for i in model.C) == 1

model.RestriccionAsignacionVehiculo = Constraint(model.V, rule=restriccion_asignacion_vehiculos)
```

Figure 14: Restricción asignación de vehículos.

Restricción para que el rango útil de los vehículos, o sea, la distancia máxima que un vehículo puede recorrer antes de quedarse sin combustible o requerir mantenimiento.

```
def restriccion_rango(model, l):
    return sum(model.DIST[i, j] * model.x[i, j, l] for i in model.C for j in model.K) <= model.R[l]

model.RestriccionRango = Constraint(model.V, rule=restriccion_rango)
```

Figure 15: Restricción rango útil del vehículo.

Restricción que activa una ruta solo si el vehículo parte del centro de distribución correspondiente.

```
def link_xy_rule(m, i, j, l):
    return m.x[i, j, l] <= m.y[i, l]
model.link_xy = Constraint(model.C, model.K, model.V, rule=link_xy_rule)
```

Figure 16: Restricción ruta desde centro de distribución.

Restricción para entregar carga a un cliente solo si hay una ruta activa hacia ese cliente.

```
def link_ux_rule(m, j, l):
    return m.u[j, l] <= m.Q[l] * sum(m.x[i, j, l] for i in m.C)
model.link_ux = Constraint(model.K, model.V, rule=link_ux_rule)
```

Figure 17: Restricción entrega de carga a un cliente.

Restricción para que cliente sea visitado solo una vez.

```
def visita_rule(m, j):
    return sum(m.x[i, j, l] for i in m.C for l in m.V) == 1
model.visita = Constraint(model.K, rule=visita_rule)
```

Figure 18: Restricción visita a cliente una sola vez.

Restricción encargada de que cada vehículo parta desde un único centro de distribución.

```
def origen_rule(m, l):
    return sum(m.y[i, l] for i in m.C) == 1
model.un_origen = Constraint(model.V, rule=origen_rule)
```

Figure 19: Restricción salida desde un único centro de distribución.

2.4 Resolver el modelo

Finalmente se resuelve el modelo llamando el solver glpk, para luego imprimir el estado de la solución y el valor óptimo de la función objetivo.

```python
solver = SolverFactory('glpk')
# Resolver
result = solver.solve(model, tee=True)

# Mostrar el estado y el valor de la función objetivo
print("Estado:", result.solver.status)
print("Óptimo:", value(model.OBJ))
```

Figure 20: Usar el solver.

2.5 Visualizacion y hallazgos principales

Foto caso 1

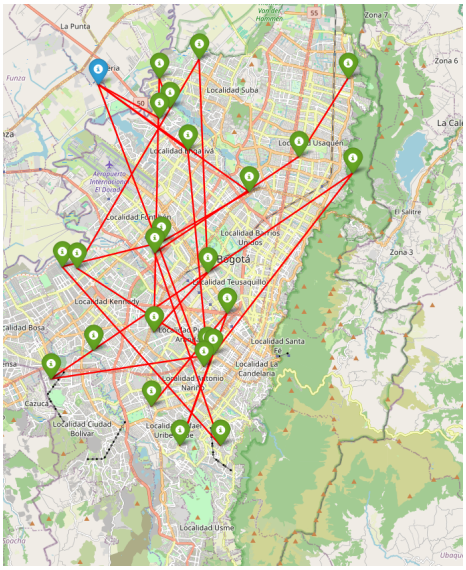

Figure 21: Caso 1

Foto caso 2

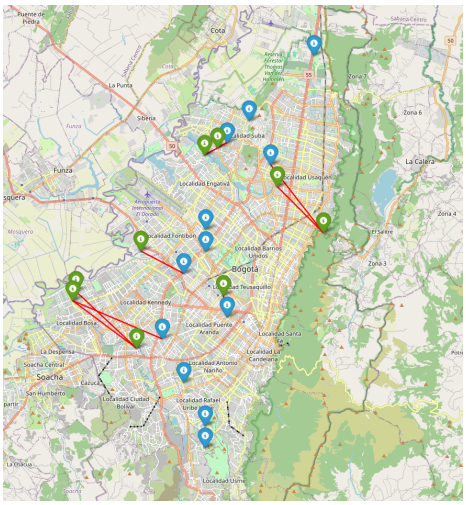

Figure 22: Caso 2

Foto caso 3

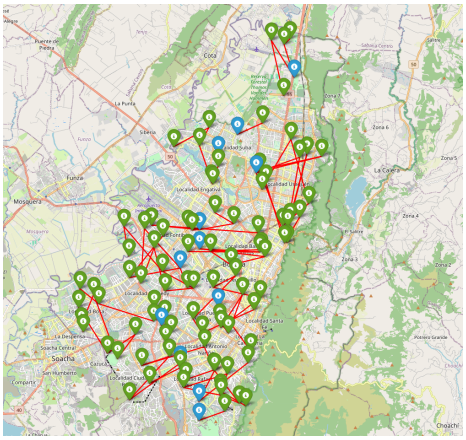

Figure 23: Caso 3

2.6 Cambios en la formulacion matematica y trade-off

Los cambios realizados en la formulacion matematica fueron realizadas en las restricciones agregando 4 restricciones nuevas para que el modelo tenga coherencia

- **Relación entre variables x_{ijl} y y_{il}:**

$$x_{ijl} \leq y_{il} \quad \forall i \in C,\ j \in K,\ l \in V \tag{1}$$

Esta restriccion asegura que un vehiculo l solo puede viajar desde un centro de distribucion i a un cliente j, lo que garantiza coherencia en la asignacion de un vehiculo a un centro de distribucion y las rutas que realiza desde ese centro

- **Límite de carga entregada por un vehículo a un cliente:**

$$u_{jl} \leq Q_l \sum_{i \in C} x_{ijl} \quad \forall j \in K,\ l \in V \tag{2}$$

Esto garantiza que la cantidad entregada a un cliente j por un vehiculi l solo puede ser mayor a 0 si el vehiculo l visita al cliente j desde algun centro de distribucion i, tambien asegura que la carga entregada no supere la capacidad del vehiculo Q_1

- **Cada cliente debe ser visitado exactamente una vez:**

$$\sum_{i \in C} \sum_{l \in V} x_{ijl} = 1 \quad \forall j \in K \tag{3}$$

Esta garantiza que cada cliente j sea visitado solo 1 vez desde cualquier centro de distribucion i y por cualquier vehiculo l. Esto cumple que la demanda de cada cliente sea cubierta completamente

- **Cada vehículo parte desde un único centro de distribución:**

$$\sum_{i \in C} y_{il} = 1 \quad \forall l \in V \tag{4}$$

Esta restricción garantiza que cada vehículo l esté asignado a un solo centro de distribución i como punto de partida. Esto significa que un vehículo no puede partir de múltiples centros ni quedarse sin un centro de origen.

2.6.1 Trade Off

Simplificación del modelo sin impacto significativo. Aumentar el número de clientes y centros de distribución no tuvo mayor problema, ya que el modelo esta diseñado enfocándose en las partes críticas del problema. Estas simplificaciones permitieron manejar la escalabilidad sin comprometer la calidad de la solución.

Omisión del retorno al centro de distribución. No se incluyó el trayecto de retorno de los vehículos al centro de distribución en el mapa ni en los archivos de verificación. Esta decisión se justificó bajo el supuesto de que cada vehículo retorna al mismo centro desde el que partió, y que dicho retorno no tiene un impacto crítico en el análisis de este caso de estudio ya que no es lo que se busca en el análisis.

Impacto: Al omitir el retorno, se redujo la carga computacional del modelo, simplificando el cálculo de distancias y costos. No obstante, esto implica una pérdida de realismo, ya que en un entorno operativo real, el vehículo podría retornar por una ruta más corta o distinta a la utilizada para atender a los clientes.

Limitación: La falta de información sobre rutas alternativas más cortas para el retorno dificulta una inclusión de manera realista. Trazar una línea recta entre el último cliente y el centro de distribución no es representativo de las condiciones reales de tráfico y vialidad.

2.7 Análisis de Sensibilidad y Reportes

2.7.1 Análisis de los diferentes casos

2.7.2 Análisis exclusivo del caso 3

- Impacto en variaciones
  Para ver el impacto en las variaciones de los valores de costos de combustible, capacidad de los centros de distribución y demandas de los clientes se hacen variaciones en los valores de estas variables para ver como cambia el costo total, los vehículos usados y el tiempo promedio.

  Los valores de la demanda de los clientes se cambiaron al azar en valores de 14, 13,15 y 20. Por otro lado los valores de capacidad de los centros de distribución se cambiaron tambien al azar por valores de 95, 200, 300 y 150.

  En la siguiente tabla se pueden ver las variaciones hechas a los valores de los combustibles:

	Costo de combustible (Pf) +20%	**Costo de combustible (Pf) -20%**
Valores iniciales	123,12	123,12
Variación	20%	-20%
Valor modificado	147,744	98,496

A continuación se ven los valores obtenidos de las variaciones

Escenario	**Base**	**Pf +20%**	**Pf -20%**
Costo total (COP)	475,596,99	482,711,356	468,482,624
Vehículos usados	25	26	28
Tiempo prom. por vehículo (min)	17.24	14.45	14.45
Carga total entregada (unid.)	298	345	345
Distancia total (km)	288.92	288.92	288.92

Table 1: Comparación de escenarios con variación en el costo de combustible (Pf)

- Identificar los parámetros que generan mayor impacto en el costo total y estructura de rutas

  De acuerdo al análisis anterior de las variaciones de los parametros, se puede decir que el parámetro que genera mayor impacto en el costo total y la estructura de rutas es la demanda de los clientes, porque al modificarlo cambia los vehículos usados y también las rutas. Además de que también cambia el tiempo promedio por vehículo disminuyéndolo en 2 minutos. Por otro lado, el precio del combustible también impacta en el costo total porque por ejemplo si lo aumentamos en un 20% el costo total aumenta 7.115 pesos y si lo disminuimos en un 20% el costo total disminuye casi en la misma cantidad de dinero 7.114. Sin embargo, estos cambios no alteran la estructura de rutas ni el número de vehículos, por lo que su impacto es económico pero no logístico.
  La carga total entregada también cambia pero es normal debido a que si hacemos variaciones en las demandas de los clientes la carga de entrega cambiará. Como parámetro con menos impacto consideramos las capacidades de los centros de distribución, ya que no influye en el costo total y ni en la estructura de las rutas.

- Reportes por centro de distribución y por vehículo
  Para hacer este informe analizaremos el archivo de verificación generado para el caso 3. Con el siguiente codigo pudimos obtener los reportes para cada centro de distribución y para cada vehículo.

```python
import pandas as pd

# Cargar archivo CSV
df = pd.read_csv('Archivos_Verificacion/verificacion_caso3.csv')  #

resumen_vehiculo = df[[
    "VehicleId", "DepotId", "TotalDistance", "TotalTime", "FuelCost", "DemandsSatisfied"
]]

print("Resumen por vehículo:")
print(resumen_vehiculo)


resumen_cd = df.groupby("DepotId").agg({
    "VehicleId": "count",
    "TotalDistance": "sum",
    "TotalTime": "sum",
    "FuelCost": "sum",
    "DemandsSatisfied": "sum"
}).reset_index().rename(columns={
    "VehicleId": "Vehículos usados",
    "TotalDistance": "Distancia total",
    "TotalTime": "Tiempo total",
    "FuelCost": "Costo total",
    "DemandsSatisfied": "Carga total entregada"
})

print("Resumen por centro de distribución:")
print(resumen_cd)
```

Figure 24: Generación de los reportes.

En la siguiente imagen se puede ver el reporte de los centros de distribución. De este reporte se puede concluir que el depósito que mas carga entregó es el depósito 8. También se puede decir que el depósito que mas vehículos usó es el 7 a pesar de que este no es el que más carga

entrega. También se puede ver que el depósito más costoso es el depósito 6 y solo entrega 34 de carga.

	DepotId	Vehículos usados	Distancia total	Tiempo total	Costo total
0	1	1	2.30	3.450	3786.0760
1	2	1	2.27	3.405	3736.6924
2	3	3	11.50	17.250	18930.3800
3	4	3	58.75	88.125	96709.5500
4	5	3	56.56	84.840	93104.5472
5	6	3	35.80	53.700	58931.0960
6	7	2	28.30	42.450	46585.1960
7	8	4	15.30	22.950	25185.6360
8	9	1	13.82	20.730	22749.3784
9	10	2	24.13	36.195	39720.8756
10	11	2	20.53	30.795	33794.8436
11	12	5	19.66	29.490	32362.7192

	Carga total entregada
0	12.0
1	12.0
2	24.0
3	36.0
4	36.0
5	34.0
6	24.0
7	48.0
8	12.0
9	24.0
10	24.0
11	12.0

Figure 25: Reporte centros de distribución

En el reporte de los vehículos se puede ver que tres vehículos no se usaron para la distribución. El vehículo 2 es el vehículo que tiene mayor costo de combustible con un valor de 87.968 y también es el vehículo que mas tiempo total tiene. Por otro lado, el vehículo 29 es el vehículo que menos combustible gastó y el que menos tiempo total tienes.

	VehicleId	DepotId	TotalDistance	TotalTime	FuelCost	DemandsSatisfied
0	1	4	38.92	58.380	64066.9904	12.0
1	2	5	53.44	80.160	87968.6528	12.0
2	3	6	7.32	10.980	12049.5984	12.0
3	4	7	4.62	6.930	7605.0744	12.0
4	5	3	6.09	9.135	10024.8708	12.0
5	6	6	26.71	40.065	43967.8652	10.0
6	7	7	23.68	35.520	38980.1216	12.0
7	8	4	18.37	27.555	30239.2244	12.0
8	9	10	18.94	28.410	31177.5128	12.0
9	10	12	19.66	29.490	32362.7192	12.0
10	11	12	0.00	0.000	0.0000	NaN
11	12	12	0.00	0.000	0.0000	NaN
12	13	11	14.54	21.810	23934.5848	12.0
13	14	12	0.00	0.000	0.0000	NaN
14	15	9	13.82	20.730	22749.3784	12.0
15	16	3	0.00	0.000	0.0000	NaN
16	17	10	5.19	7.785	8543.3628	12.0
17	18	6	1.77	2.655	2913.6324	12.0
18	19	3	5.41	8.115	8905.5092	12.0
19	20	4	1.46	2.190	2403.3352	12.0
20	21	8	4.84	7.260	7967.2208	12.0
21	22	12	0.00	0.000	0.0000	NaN
22	23	1	2.30	3.450	3786.0760	12.0
23	24	2	2.27	3.405	3736.6924	12.0
24	25	8	8.89	13.335	14634.0068	12.0
25	26	5	2.45	3.675	4032.9940	12.0
26	27	8	0.98	1.470	1613.1976	12.0
27	28	5	0.67	1.005	1102.9004	12.0
28	29	8	0.59	0.885	971.2108	12.0
29	30	11	5.99	8.985	9860.2588	12.0

Figure 26: Reporte de los vehículos.

- Estadísticas globales

Estadística	**Valor promedio**
Distancia por vehículo	11.56 km
Tiempo por vehículo	17.34 min
Carga entregada por vehículo	11.92 unidades

Table 2: Estadísticas globales promedio

Variable	**Desviación estándar**
Tiempo por vehículo	19.65 min
Distancia por vehículo	19.65 km

Table 3: Desviaciones estándar

En cuanto a la distribución de los datos de distancias por vehículos se puede ver una mayor concentración en las distancias bajas, o sea, un sesgo a la derecha. Se observa que la mayoría de los vehículos están en rangos cortos, con distancias menores a 10 km, lo cual sugiere una asignación eficiente en términos de proximidad entre los clientes y sus centros de distribución. Por otro lado, a medida que aumenta la distancia la

frecuencia de vehículos disminuye lo que indica que muy pocos vehículos están realizando rutas largas.

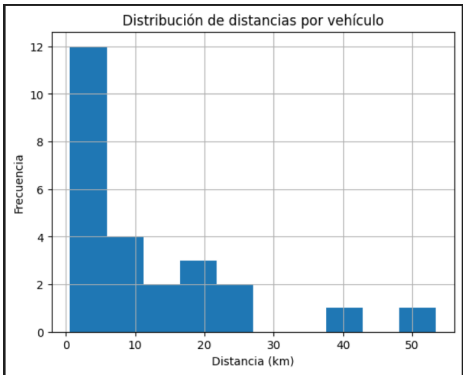

Figure 27: Distribución de los datos

- Conclusiones

  ¿Dónde se presentan los mayores cuellos de botella?

  Los principales cuellos de botella en la operación logística de LogistiCo son: - La concentración de demanda en ciertos centros de distribución

  - La flota limitada, ya que en escenarios de alta demanda, se requiere un número mayor de vehículos, lo que podría no ser viable sin una ampliación de flota o una redistribución de rutas.

  - Clientes con ubicación lejana. Algunos vehículos deben recorrer distancias mayores lo que puede llevar a sobrecostos o subutilización de su capacidad de carga.

  ¿Qué mejoras recomendaría a LogistiCo?

  - Le recomendaría expandir la flota de vehículos priorizando vehículos con mayor capacidad o autonomía para zonas alejadas. También revisar la ubicación y capacidad de los centros de distribución para balancear mejor la carga y evitar saturación de ciertos puntos.

  - Implementar sistemas de monitoreo que permitan ajustar rutas y asignación de recursos de manera dinámica.