



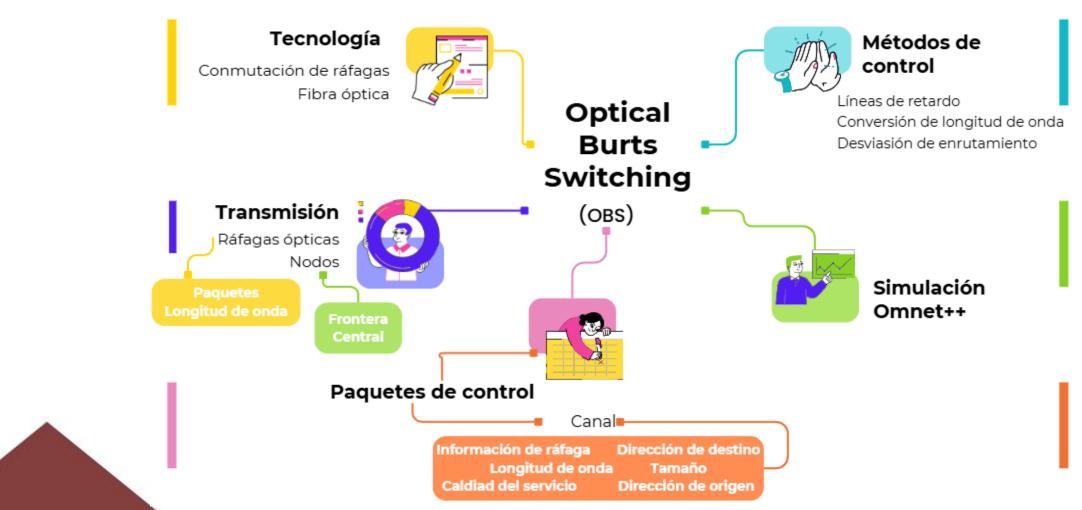
## Contenido

- Contexto de la problemática
- 2 Métodos de control
- 3 Simulación de una red de fibra óptica
- Modelo de aprendizaje
- Evaluación y Resultados
- 6 Limitaciones del modelo
- 7 Actividad



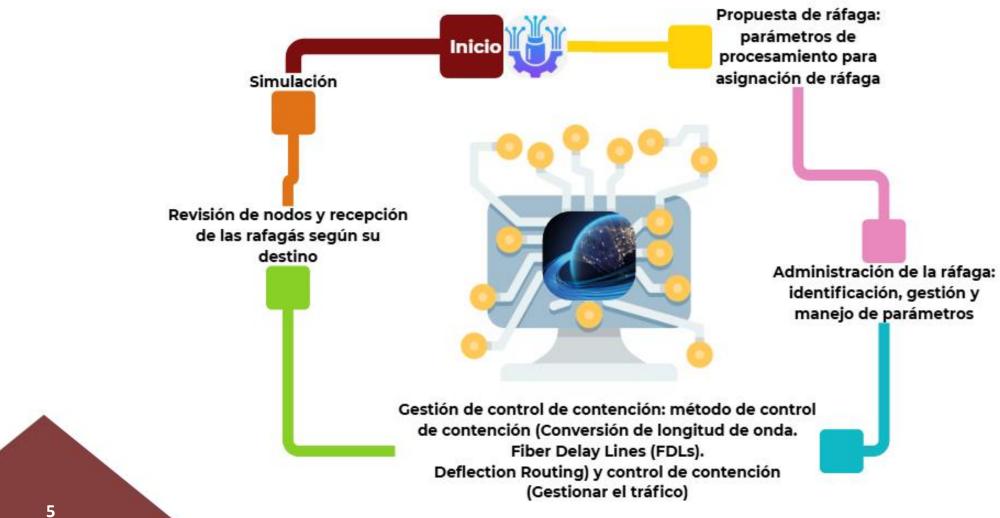


## 1. Contexto de la problemática



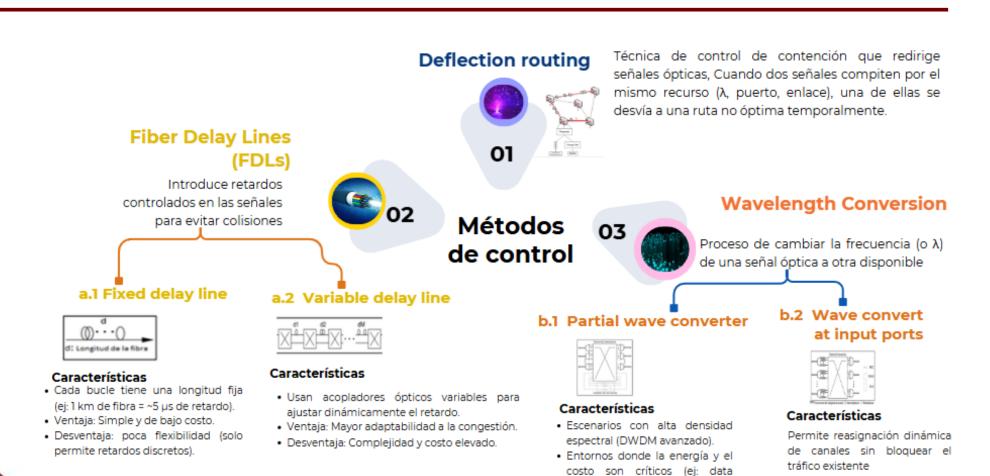
## 2. Métodos de control





## 2. Métodos de control

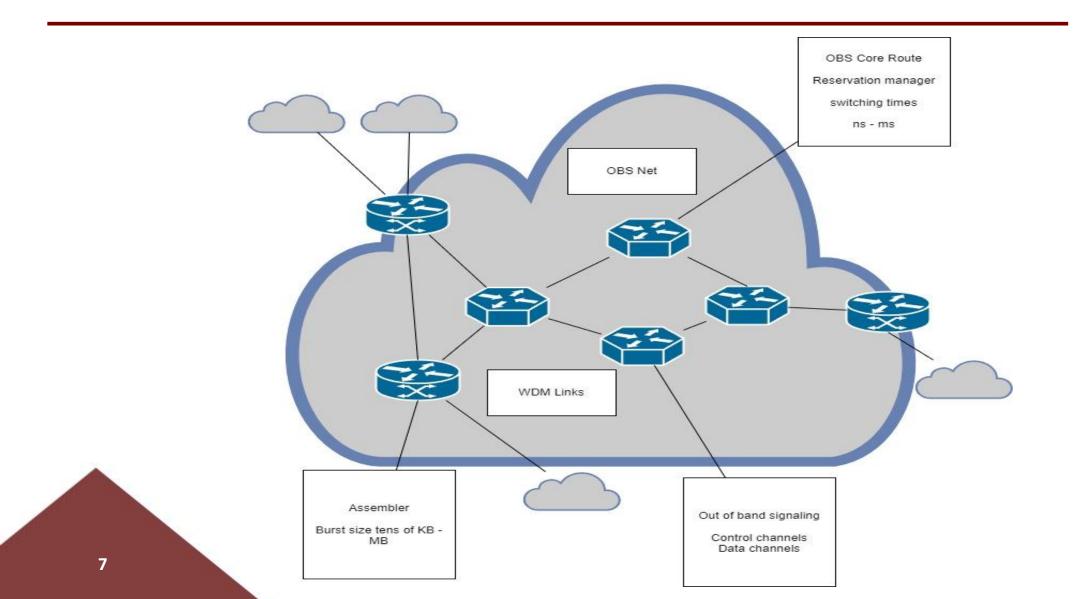


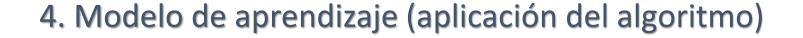


centers).



## 3. Simulación de una red fibra óptica en OMNeT++

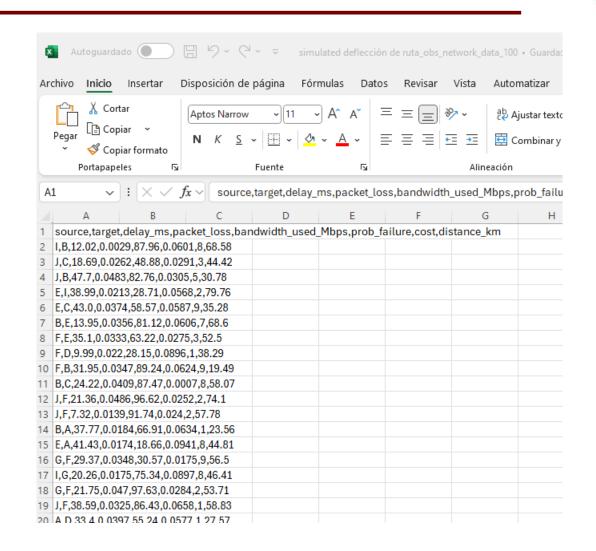






Archivo .csv con 100 líneas de datos simulados de deflexión de ruta para una OBS en OMNeT++ con las siguientes columnas:

- •source
- target
- •delay\_ms
- packet loss
- •bandwidth\_used\_Mbps
- •prob\_failure
- •cost
- •distance\_km



## 4. Modelo de aprendizaje (aplicación del algoritmo)



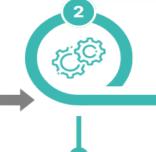


#### Instalación de librerías

!pip install -q polars networkx matplotlib

import polars as pl import networks as nx import matplotlib.pyplot as plt import pandas as pd

from google.colab import files



#### Subir archivo desde dispositivo

print(" 11 Carga tu archivo CSV con columnas: source,target,delay\_ms,packet\_loss,cost,distance \_km,prob\_failure") uploaded = files.upload()

file\_name = list(uploaded.keys())[0] df = pl.read\_csv(file\_name)

raise ValueError("X No se cargó ningún archivo")



#### Crear grafo dirigido con atributos

G = nx.DiGraph()

for row in df.to\_dicts(): G.add\_edge(row['source'], row['target'], delay=row['delay\_ms'], packet\_loss=row['packet\_loss'], cost=row['cost'], distance=rowl'distance km'l. prob\_failure=row['prob\_failure'])

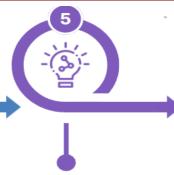


#### Algoritmo de Dijkstra

def dijkstra(graph, start, end, weight='weight'): path = nx.dijkstra\_path(graph,

source=start, target=end, weight=weight) nx.dijkstra\_path\_length(graph, source=start, target=end, weight=weight)

return path, total\_weight



#### Visualización del grafo con más rutas

def draw\_graph\_with\_path(graph, path, metric): pos = nx.spring\_layout(graph, seed=42) plt.figure(figsize=(12, 8))

# Dibujar todos los nodos y todas las aristas nx.draw\_networkx\_nodes(graph, pos, node\_color='lightblue', node\_size=800) nx.draw\_networkx\_labels(graph, pos, font\_size=10, font\_weight='bold')

# Dibujar todas las aristas en gris nx.draw\_networkx\_edges(graph, pos, edge\_color='gray', width=1.5)

# Resaltar aristas del camino óptimo if path:

edge\_path = list(zip(path[:-1], path[1:])) nx.draw\_networkx\_edges(graph, pos, edgelist=edge\_path, edge\_color='red', width=3)

# Etiquetas con valor de la métrica elegida edge\_labels = nx.get\_edge\_attributes(graph, metric) nx.draw\_networkx\_edge\_labels(graph, pos, edge\_labels={(u, v): f"{d:.2f}" for (u, v), d in edge\_labels.items()},

font\_color='black', font\_size=8)

plt.title(f"Ruta óptima resaltada usando la métrica: {metric}") plt.axis('off') plt.show()



#### Ejecutar para múltiples métricas

start\_node = 'A' end\_node = 'F' metrics = ['delay', 'packet\_loss', 'cost', 'distance', 'prob\_failure']

for metric in metrics:

path, total = dijkstra(G, start\_node, end\_node, weight=metric)

print(f"\n € Ruta óptima de '{start\_node}' a '{end\_node}' usando '{metric}':")

print(" → ".join(path)) print(f" Total {metric}: {total:.4f}")

draw\_graph\_with\_path(G, path, metric) except nx.NetworkXNoPath:

print(f" No hay ruta de {start\_node} a {end\_node}

except Exception as e:

print(f" A Error con métrica '{metric}': {e}')

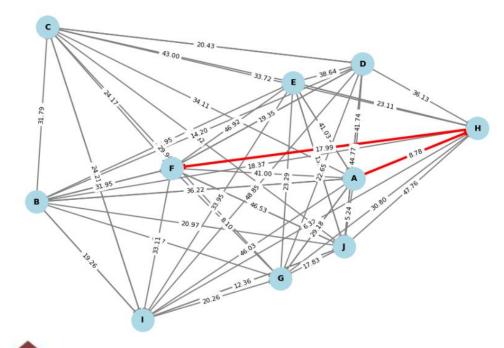
## 5. Evaluación y Resultados



Ruta óptima de 'A' a 'F' usando 'delay':

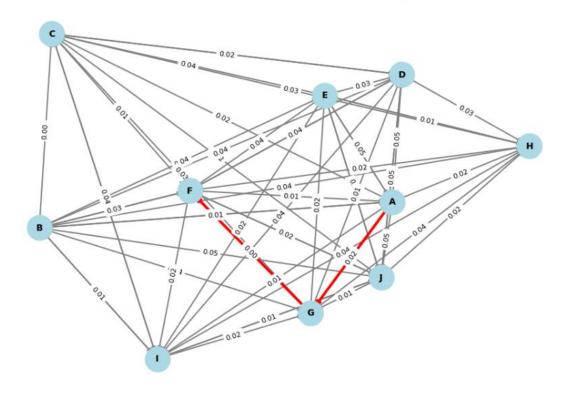
■ Total delay: 26.7700

Ruta óptima resaltada usando la métrica: delay



Ruta óptima de 'A' a 'F' usando 'packet\_loss': Total packet\_loss: 0.0160

Ruta óptima resaltada usando la métrica: packet\_loss

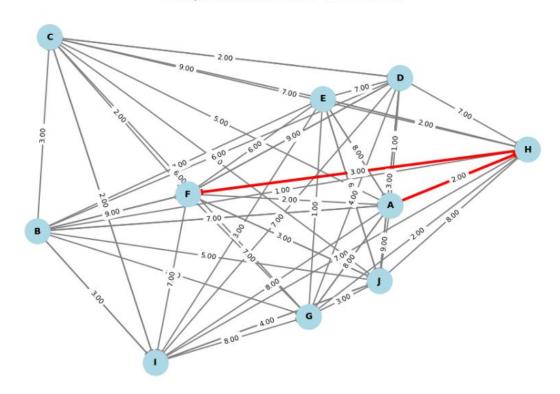






```
\bigcirc Ruta óptima de 'A' a 'F' usando 'cost': A \rightarrow H \rightarrow F \bigcirc Total cost: 5.0000
```

Ruta óptima resaltada usando la métrica: cost



## 5. Evaluación y Resultados

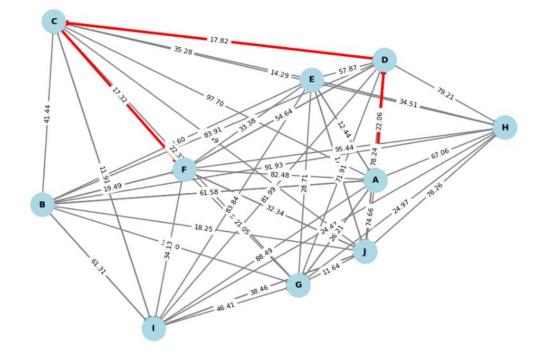


Ruta óptima de 'A' a 'F' usando 'distance':

 $A \rightarrow D \rightarrow C \rightarrow F$ 

Total distance: 57.2000

Ruta óptima resaltada usando la métrica: distance

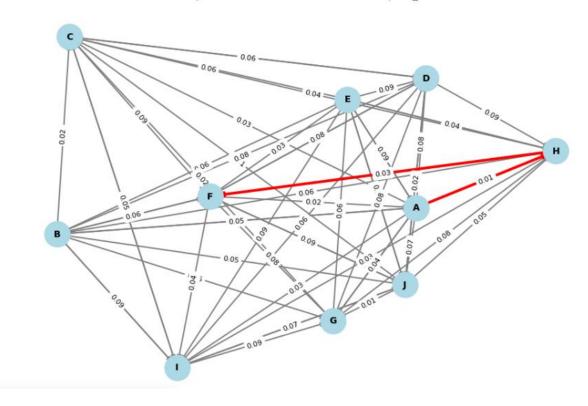


Ruta óptima de 'A' a 'F' usando 'prob\_failure':

 $A \rightarrow H \rightarrow F$ 

Total prob\_failure: 0.0413

Ruta óptima resaltada usando la métrica: prob\_failure



### 7. Actividad



Análisis de datos de deflexión de ruta, con un enfoque en la optimización del uso de energía en el envío de paquetes.



# Gracias!!! jortiza@udistrital.edu.co



# Falta implementar el aprendizaje supervisado



Proyecto final: randomforest con criterio, entonces hacer el paso a paso (crear el algoritmo) para reemplazar criterio