# PARCIAL

# INTRODUCCION A LA INTELIGENCIA ARTIFICIAL


## NOMBRE

## DADY SNEIDER LOAIZA LOAIZA

## PROFESOR

## JAVIER ALBERTO OCHOA AVILA


## GRUPO

## #01

Para resolver el problema del viajero ambulante, que consiste en recorrer 9 ciudades partiendo desde las 10 a. m., comencé preguntándome cómo podría encontrar la ruta óptima. Sin embargo, para lograrlo, primero necesitaba determinar los grafos

correspondientes. Utilicé una matriz de adyacencia, ya que esta me permitía representar las conexiones entre las ciudades de manera clara y facilitaba su uso en el proceso de búsqueda de la mejor ruta.

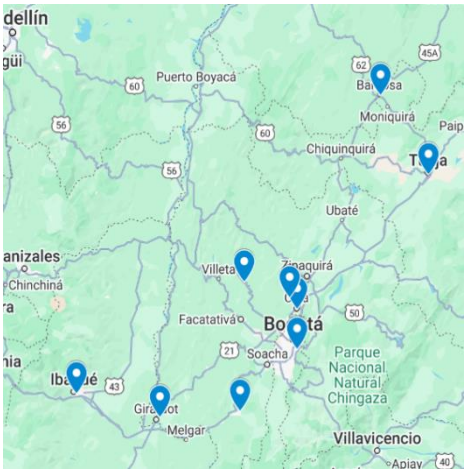
DISTANCIA

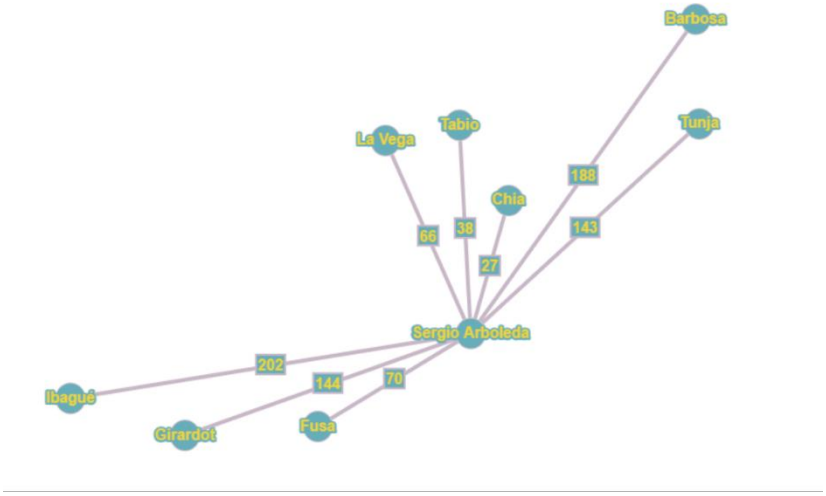
| Ciudad | USA | Tunja | Chia | La vega | Fusa | Girardot | Tabio | Ibagué | Barbosa |
|---|---|---|---|---|---|---|---|---|---|
| **USA** | 0km | 143km | 27km | 66km | 70km | 144km | 38km | 202km | 188km |
| **Tunja** | 143km | 0km | 124km | 181km | 222km | 268km | 161km | 341km | 48km |
| **Chia** | 27km | 124km | 0km | 52km | 94km | 145km | 21km | 223km | 138km |
| **La vega** | 66km | 181km | 52km | 0km | 83km | 134km | 62km | 205km | 157km |
| **Fusa** | 70km | 222km | 94km | 83km | 0km | 94km | 103km | 176km | 212km |
| **Girardot** | 144km | 268km | 145km | 134km | 94km | 0km | 151km | 127km | 261km |
| **Tabio** | 38km | 161km | 21km | 62km | 103km | 151km | 0km | 232km | 147km |
| **Ibagué** | 202km | 341km | 223km | 205km | 176km | 127km | 232km | 0km | 315km |
| **Barbosa** | 188km | 48km | 138km | 157km | 212km | 261km | 147km | 315km | 0km |

TIEMPO

| Ciudad | USA | Tunja | Chia | La vega | Fusa | Girardot | Tabio | Ibagué | Barbosa |
|---|---|---|---|---|---|---|---|---|---|
| **USA** | 0h 0m | 2h 30m | 45m | 1h 30m | 2h | 3h | 1h | 4h | 3h 30m |
| **Tunja** | 2h 30m | 0h 0m | 2h | 3h | 4h | 5h | 3h | 6h | 1h |
| **Chia** | 45m | 2h | 0h 0m | 1h | 1h 30m | 2h 30m | 1h | 4h | 2h |
| **La vega** | 1h 30m | 3h | 1h | 0h 0m | 1h 30m | 2h | 1h 30m | 3h 30m | 3h |
| **Fusa** | 2h | 4h | 1h 30m | 1h 30m | 0h 0m | 1h 30m | 2h | 3h | 4h |
| **Girardot** | 3h | 5h | 2h 30m | 2h | 1h 30m | 0h 0m | 2h 30m | 2h 30m | 5h |
| **Tabio** | 1h | 3h | 1h | 1h 30m | 2h | 2h 30m | 0h 0m | 4h | 2h 30m |
| **Ibagué** | 4h | 6h | 4h | 3h 30m | 3h | 2h 30m | 4h | 0h 0m | 5h 30m |
| **Barbosa** | 3h 30m | 1h | 2h | 3h | 4h | 5h | 2h 30m | 5h 30m | 0h 0m |

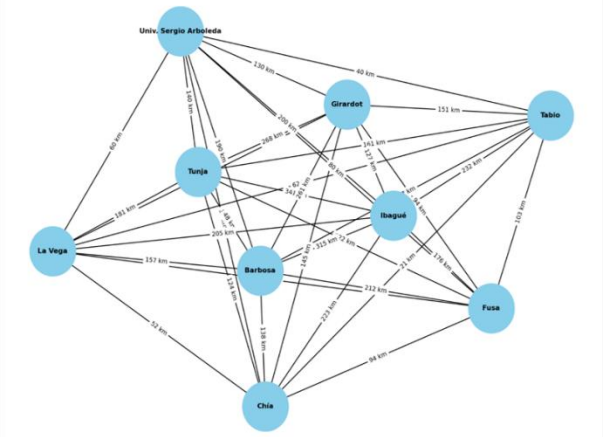
Luego comencé a planificar mis grafos usando como referencia un mapa que trazaba las rutas de manera gráfica, para poder intuir distintas rutas.




Me preguntaba si, partiendo desde lo más lejano, podría llegar a una buena ruta o, por el contrario, si sería mejor comenzar desde lo más cercano. Decidí iniciar con lo más cercano, utilizando un algoritmo de fuerza bruta que probaba numerosas variaciones. Así, me di cuenta de que existían rutas más cortas y otras más largas, pero no lograba determinar cuál sería la mejor. Entonces, opté por descartar las opciones menos óptimas, quedándome solo con las rutas más prometedoras. Mi método combinaba el algoritmo de fuerza bruta, que prueba todas las posibilidades hasta encontrar la más favorable, con un algoritmo de ramificación y poda, que me permitía eliminar las opciones menos eficaces, haciendo mi búsqueda mucho más óptima.

Pero, ¿cómo lograr esto? No podía hacerlo a lápiz y papel, ni escribiendo, ya que sería extremadamente tardado. Por ello, recurrí a la inteligencia artificial y solicité un código que me permitiera explorar todas las posibles variaciones, como las del siguiente grafo:
distancias = { 'USA': {'Tunja': 143, 'Chia': 27, 'La Vega': 66, 'Fusa': 70, 'Girardot': 144, 'Tabio': 38, 'Ibagué': 202, 'Barbosa': 188},
'Tunja': {'USA': 143, 'Chia': 124, 'La Vega': 181, 'Fusa': 222, 'Girardot': 268, 'Tabio': 161, 'Ibagué': 341, 'Barbosa': 48}, 'Chia': {'USA':



```
27, 'Tunja': 124, 'La Vega': 52, 'Fusa': 94, 'Girardot': 145, 'Tabio': 21, 'Ibagué': 223, 'Barbosa': 138},  'La Vega': {'USA': 66, 'Tunja': 181, 'Chia': 52, 'Fusa': 83, 'Girardot': 134, 'Tabio': 62, 'Ibagué': 205, 'Barbosa': 157},'Fusa': {'USA': 70, 'Tunja': 222, 'Chia': 94, 'La Vega': 83, 'Girardot': 94, 'Tabio': 103, 'Ibagué': 176, 'Barbosa': 212},  'Girardot': {'USA': 144, 'Tunja': 268, 'Chia': 145, 'La Vega': 134, 'Fusa': 94, 'Tabio': 151, 'Ibagué': 127, 'Barbosa': 261},'Tabio': {'USA': 38, 'Tunja': 161, 'Chia': 21, 'La Vega': 62, 'Fusa': 103, 'Girardot': 151, 'Ibagué': 232, 'Barbosa': 147},'Ibagué': {'USA': 202, 'Tunja': 341, 'Chia': 223, 'La Vega': 205, 'Fusa': 176, 'Girardot': 127, 'Tabio': 232, 'Barbosa': 315},  'Barbosa': {'USA': 188, 'Tunja': 48, 'Chia': 138, 'La Vega': 157, 'Fusa': 212, 'Girardot': 261, 'Tabio': 147, 'Ibagué': 315}}

ciudad_actual = 'USA'

ciudades_visitadas = [ciudad_actual]

distancia_total = 0

print("Bienvenido al recorrido de ciudades. Comenzamos en USA.")

while len(ciudades_visitadas) < len(distancias):
    print(f"\nEstás en {ciudad_actual}. Ciudades visitadas: {', '.join(ciudades_visitadas)}")
    ciudades_disponibles = [ciudad for ciudad in distancias[ciudad_actual].keys() if ciudad not in ciudades_visitadas]
    if not ciudades_disponibles:
        print("No hay más ciudades disponibles para visitar.")
        break
    print("Ciudades disponibles para visitar:")
    for i, ciudad in enumerate(ciudades_disponibles, 1):
        print(f"{i}. {ciudad} ({distancias[ciudad_actual][ciudad]} km)")
    try:
        eleccion = int(input("Elige el número de la ciudad a la que quieres ir: ")) - 1
        if eleccion < 0 or eleccion >= len(ciudades_disponibles):
            print("Elección no válida. Intenta de nuevo.")
            continue
    except ValueError:
        print("Entrada no válida. Por favor, ingresa un número.")
        continue
    ciudad_elegida = ciudades_disponibles[eleccion]
    distancia = distancias[ciudad_actual][ciudad_elegida]
    distancia_total += distancia
    ciudades_visitadas.append(ciudad_elegida)
    ciudad_actual = ciudad_elegida
distancia_final = distancias[ciudad_actual]['USA']

distancia_total += distancia_final

ciudades_visitadas.append('USA')

print("\nRecorrido completado.")

print(f"Ciudades visitadas en orden: {', '.join(ciudades_visitadas)}")

print(f"Distancia total recorrida: {distancia_total} km")
```

Gracias a este análisis, puedo concluir que la ruta óptima es: **Universidad Sergio Arboleda → Tabio → Chía → Tunja → Barbosa → La Vega → Fusa → Girardot → Ibagué → Universidad Sergio Arboleda**, con un recorrido estimado de **894 km**. Sin embargo, aunque la distancia es eficiente, el tiempo no es el óptimo, ya que este recorrido tarda aproximadamente **18 horas y 55 minutos**.

Fue un proceso largo y desafiante. Al principio, mis conceptos no eran muy claros, especialmente en lo relacionado con los algoritmos y los grafos. No obstante, poco a poco fui comprendiendo mejor el tema, lo que facilitó la resolución de este problema hasta llegar a esta conclusión.

1. INICIO
2. Definir N, i, j, ciudad Inicio como Entero
3. Definir matriz Distancias como Matriz de Enteros
4. Definir mejor Ruta como Lista de Enteros
5. Definir distancia Mínima como Entero
6. ciudad Inicio ← "USA"
7. distancia Mínima ← Infinito
8. Escribir "Ingrese el número de ciudades:"
9. Leer N
10. Si N < 1 Entonces
11. Escribir "El número de ciudades debe ser mayor que 0."
12. Terminar
13. Fin Si
14. Para i ← 1 Hasta N Hacer
15. Para j ← 1 Hasta N Hacer
16. Si i ≠ j Entonces
    a. Escribir "Ingrese la distancia entre ciudad ", i, " y ciudad ", j, ":"
    b. Leer matriz Distancias[i][j]
17. Fin Si
18. Fin Para
19. Fin Para
20. Para i ← 1 Hasta N Hacer
21. Si matriz Distancias["USA"][i] < distancia Mínima Entonces
22. Distancia Minima ← matrizDistancias["USA"][i]
23. Mejor Ruta ← [i]
24. Fin Si
25. Fin Para
26. Escribir "La mejor ruta desde USA es: ", mejor Ruta
27. Escribir "Distancia mínima: ", distancia mínima
28. FIN