El camino al menor número de transbordos y costo de pasajes

Integrantes:
Paola Andrea Campiño
Entrega 1:Primera aproximación del Modelo Matemático
Modelado, Simulación y Optimización

Departamento de Ingeniería de Sistemas y Computación
Universidad de Los Andes
Bogotá, Colombia

Link al repositorio: https://github.com/Paolaaaaaa/MOS

1 Descripción del Problema

Contexto:

Bogotá es una ciudad muy grande con un sistema de transporte público complejo que abarca el Transmilenio y los obuses del SITP. Imagina que un día necesitas viajar desde tu hogar a un lugar que nunca has visitado antes, por lo que careces de conocimiento sobre la ruta adecuada y de ninguna forma planeas gastar más de tu presupuesto de 7k pesos solo para llegar al lugar.

El desafío surge al buscar la manera más eficiente de llegar a tu destino, minimizando tanto el tiempo de viaje como las complicaciones. Esto implica encontrar la combinación óptima de rutas de Transmilenio y paradas del SITP para ir desde tu estación de origen hasta la estación de destino con el menor número posible de transbordos, paradas y costos.

El desafío radica en proporcionar a los usuarios una herramienta o aplicación que, dados su punto de partida y su destino, calcule automáticamente la ruta más eficiente, el número de transbordos necesarios y las paradas intermedias . Esto permitiría a los pasajeros tomar decisiones informadas para sus desplazamientos diarios en la ciudad.

Variables a minimizar:

- El número de transbordos.
- El número de paradas.
- El costo total del viaje

Limitaciones y restricciones del problema:

- Hay una estación (nodo) de origen.
- Hay una estación (nodo) de destino.

– Hay estaciones (nodos) intermedios que tienen un estación (nodo) de llegada y una estación (nodo) de salida.
– El número de transbordos está definido como el hecho de usar más de una ruta para llegar al destino
– El costo del viaje no puede sobrepasar un límite

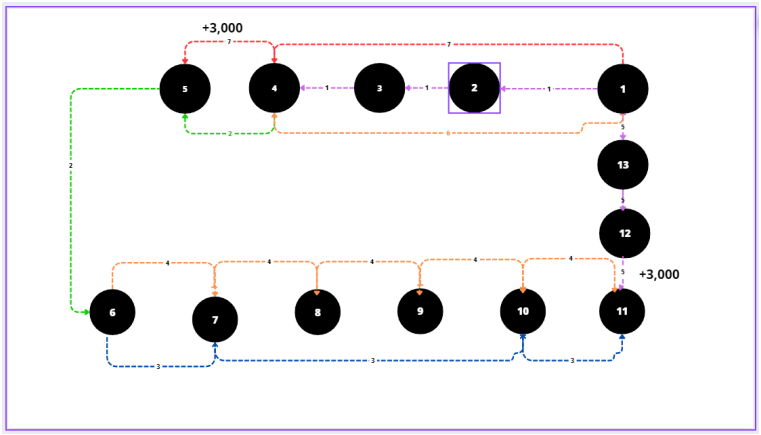

Fig. 1. Representación grafica del escenario 1

2 Conjuntos, Parámetros y Variables

parámetros y variables de decisión que se requieren para plantear el modelo matemático:

Table 1. Conjuntos, Parámetros y Variables de decisión.

Sets and Parameters	**Description**
N	Set de estaciones (node) .
R	Set de rutas .
o	Estación origen (nodo).
d	Estación destino (nodo).
C	Costo maximo a gastar.
$rutas_{ijk}$	Determina la existencia de un enlace entre dos estaciones (i y j) puede tomar los valores 1 o 999. t hace refencia a el bus/ruta existente entre los dos puntos
$costo_{ijk}$	Determina el costo de pasar entre estaciones en caso de darse un camino costo va a tomar el costo de pasar se está trabjando un costo de 3 . Que sería 3 mil pesos en una situación real

Table 2. Variables de decisión

Variables	Description
X_{ijk}	Determina el enlace para moverse entre estaciones (i y j) si va a ser usado puede tomar valores 1s o 0s. 0 para no usar esa ruta y 1 para usar esa ruta. t hace referencia a el bus/ruta
Y_k	Determina las rutas que se han usado durante el trayecto. La variable k hace referencia a la ruta usada.

3 Función Objetivo y Restricciones (50%)

Expresión Matemática de la Función Objetivo y Restricciones

Definiciones:

- Transbordo: Es cambiar de ruta
- Parada: Es la cantidad estaciones visitadas para llegar al destino. Esto incluye la para de destino.
- Siempre se considera un costo inicial, ya que se asume que el usuario se encuentra dentro del transporte (incluso si se encuetra en una para de autobus).

Función Objetivo (F.O)

La función objetivo busca minimizar el número de transbordos, el número de paradas y costo total de viajar.

Minimizar: $F.O = \alpha$·paradas+β·Número de transbordos+ϵ·(3+costos de viajar por ruta)

Esto puede ser expresarse con la siguente expresión matematica:

$$\begin{aligned} \min\Bigg\{ & \alpha \sum_{i\in N}\sum_{j\in N}\sum_{k\in R}(X_{i,j,k}\cdot rutas_{i,j,k}) \\ & + \beta \sum_{k\in R}(Y_k) \\ & + \epsilon\left(3+\sum_{i\in N}\sum_{j\in N}\sum_{k\in R}(X_{i,j,k}\cdot rutas_{i,j,k}\cdot costo_{i,j,k})\right)\Bigg\} \end{aligned} \tag{1}$$

Donde:
- α, ϵ y β, son pesos que se pueden ajustar para dar importancia relativa a cada uno de los tres objetivos (minimizar paradas, costos total de transporte y la minimización de transbordos y). Los valores pueden ser seleccionados en base a las preferencias del usuario final.

- Cabe resaltar que el costo de viajar por ruta toma en cuenta una suma a 3 obligatoria, ya que se asume que la persona ya está dentro de el sistema de transporte.

Restricciones

1. Origen fijo

$$\sum_{i,j,k|i=o} X_{i,j,k} = 1 \quad \text{para el nodo origen hay solo un enlace/bus de origen}$$

2. Destino fijo

$$\sum_{j,j,k|i=d} X_{j,i,k} = 1 \quad \text{para el destino hay solo un enlace/bus de destino}$$

3. Nodos intermedios

$$\sum_{i,j,k|i!=d \ \& \ i!=o} X_{i,j,k} = \sum_{j,i,k|!=d \ \& \ i!=o} X_{j,i,k} \quad \text{Hay nodos intermedios}$$

4. Cantidad de rutas usadas:

$$\forall_{i,j,k} \quad Y_k \geq X_{ijk} \text{ Detector de rutas usadas}$$

5.Costo maximo del viaje:

$$\sum_{i \in N} \sum_{j \in N} \sum_{k \in R} X_{i,j,k} \cdot costo_{i,j,k} + 3 \leq C \quad \text{Costo total tiene que menor a } c$$

4 Implementación y resultados del Modelo Matemático

4.1 Escenario 1:

En este primer escenario (Fig. 1) se pueden ver las rutas, nodos y nodos representando las estaciones. Asimismo, hay ciertas conexiones que tienen un costo, el cual se puede observar mencionado como +3000:

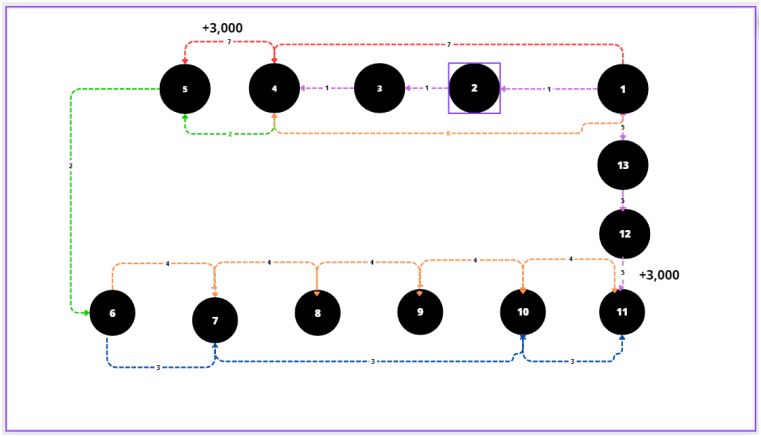

Fig. 2. Mapa escenario 1

1. Prueba de Priorización de minimizar transbordos En esta primera prueba con el escenario 1 se presenta la necesidad de desplazarse desde el nodo 1 hasta el nodo 5. Existen varias rutas para llegar, pero algunas requieren de 1 a 2 transbordos. En ocasiones, esto implica un costo adicional debido al cambio entre modalidades de transporte, como SITP o Transmilenio, con un precio estándar de 3 mil. Considerando la importancia de minimizar el costo (0.1), los transbordos (0.8) y la cantidad de paradas (0.1), se muestra esta situación en la imagen adjunta (fig 3), donde se espera que únicamente se utilice la ruta 7 por lo tanto que se den 0 transbordos, un costo total de 6 mil pesos y 2 paradas.

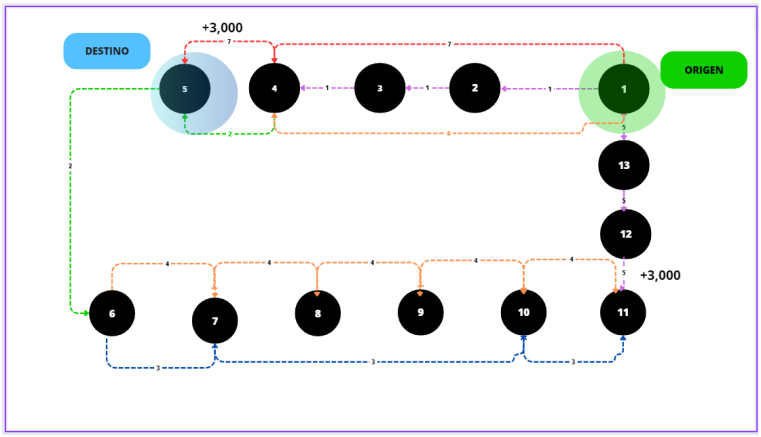

Fig. 3. Prueba 1: Priorizar minimizar transbordos

Resultados del escenario 1 prueba 1: Como se ha mencionado anteriormente se ha priorizado transbordos por lo que funciones objetivo como lo son costo o paradas no van a tener relevancia. Esto en efecto se puede ver en la imagen (fig 4) generada por el código entregado con el nombre proyecto_2_esc1_1.py Al final se puede ver el siguiente trayecto como la solución al problema, y en efecto se priorizado el hecho de que no se den muchos transbordos.

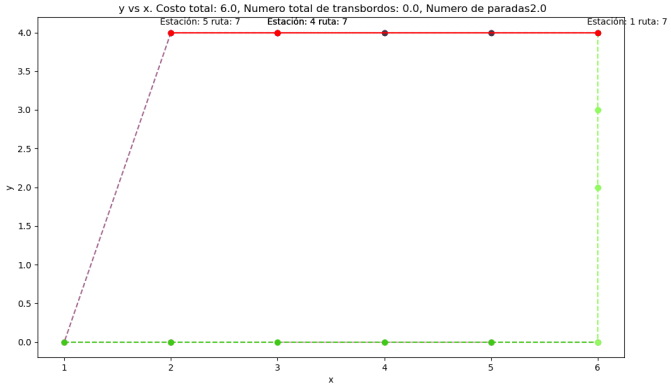

Fig. 4. Prueba 1: resultados de minimizar transbordos

Tenga en cuenta que el camino seleccionado está de color rojo

2. Priorización de minimizar Costos Considerando la importancia de minimizar el costo (0.8), los transbordos (0.1) y la cantidad de paradas (0.1). Así mismo se considera la necesidad de no gastar más de 7 mil pesos . Ahora considerando que el nodo origen es la estación 1 y el nodo destino es la estación 5. A continuación, se presenta el problema con la siguiente imagen (fig 5) se espera que únicamente se llegue a utilizar la ruta 1 y ruta 2.

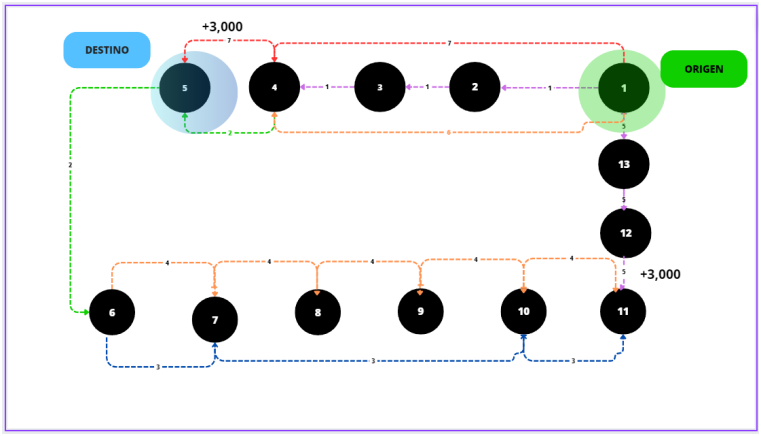

Fig. 5. Prueba 2: resultados de minimizar costos del viaje

Resultados del escenario 1 prueba 2: Se ha llegado a minimizar los costos correctamente, este se puede ver si es comparado al resultado de minimizar transbordos se ha llegado a solo tener que gastar 3 mil pesos. Sin embargo, se llevó a aumentar el número de transbordos significativamente. En la siguiente grafica (fig 6) se muestra el trayecto a tomar. cabe resaltar que para ver la ejecución puede ejecutar el archivo .py con el nombre proyecto_2_esc1_2.py

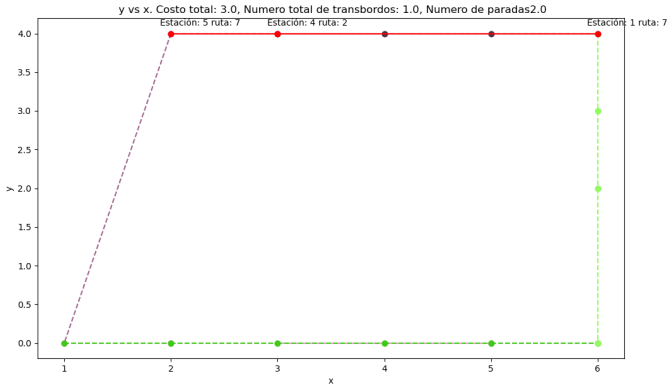

Fig. 6. Prueba 2: resultados de minimizar costos del viaje

Tenga en cuenta que el camino seleccionado está de color rojo

3. Priorización de minimizar paradas para llegar a la estación destino. Considerando el mismo mapa utilizado en el escenario uno, pero con un origen y destino diferentes, se busca minimizar el costo (0.1), los transbordos (0.1) y la cantidad de paradas (0.8). La situación se ilustra en la imagen adjunta, donde se espera utilizar exclusivamente la ruta 6 desde el nodo 1 al 4, la ruta 2 desde el 5 al 5, y del 5 al 6; y finalmente, la ruta 3 desde el nodo 6 al 7 y del 7 al 10. A continuación se puede ver una representación gráfica del problema planteado:

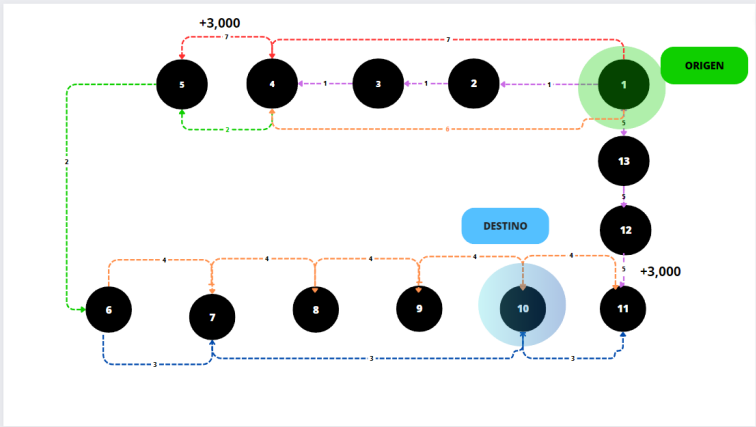

Fig. 7. Prueba 3: Planteamiento del problema de minimizar las paradas

Resultados del escenario 1 prueba 3: En efecto sigue una ruta similar a la esperada, logrando minimizar la cantidad de paradas. Para llegar al nodo 4, siguió la vía más directa. En el caso de ir del nodo 6 al nodo 10, eligió la ruta más rápida, asegurando solo 2 paradas en la ruta 3, en comparación con las 4 paradas de la ruta 4.

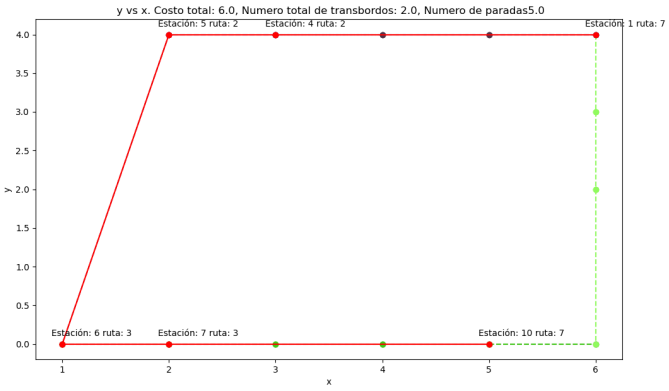

Fig. 8. Prueba 3:Resultados de minimizar las paradas

Tenga en cuenta que el camino seleccionado está de color rojo

4. Priorización de minimizar transbordos y costos llegar a la estación destino. Considerando el mismo mapa utilizado en el escenario uno, pero con un origen y destino diferentes, se busca minimizar el costo (0.45), los transbordos (0.45) y la cantidad de paradas (0.1). La situación se ilustra en la imagen adjunta, donde se espera utilizar exclusivamente la ruta 6 desde el nodo 1 al 4, la ruta 2 desde el 5 al 5, y del 5 al 6; y finalmente, la ruta 3 desde el nodo 6 al 7 y del 7 al 10. A continuación se puede ver una representación gráfica del problema planteado:

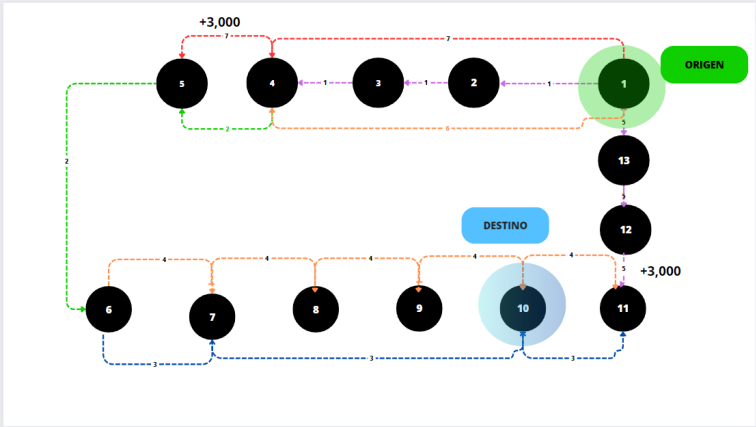

Fig. 9. Prueba 4: Planteamiento del problema de minimizar transbordos y costos

Resultados del escenario 1 prueba 4: En efecto se sigue una ruta similar a la esperada, logrando minimizar la cantidad de paradas. Para llegar al nodo 4, siguió la vía más directa, con tal de minimizar costos se hizo un transbordos con la ruta 2 así evitando la conexión que hace la ruta 7 que tine costo de 3, asegurando solo 2 paradas en la ruta 3, en comparación con las 4 paradas de la ruta 4. Así mismo puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_2_esc1_4.py"

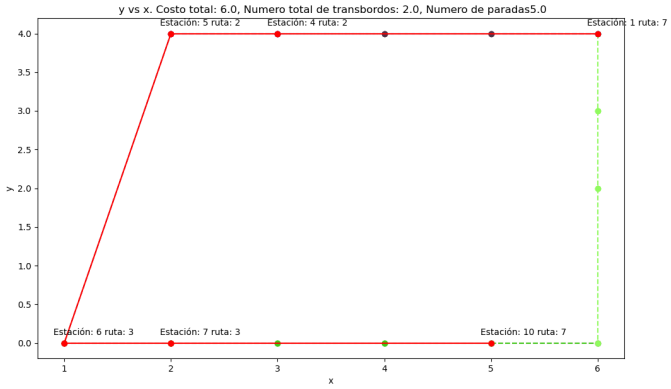

Fig. 10. Prueba 4:Resultados de minimizar las Transbordos y costos para llegar a la estación destino

Tenga en cuenta que el camino seleccionado está de color rojo

4.2 Escenario 2:

En este primer escenario (Fig. 11) se pueden ver las rutas, nodos y nodos representando las estaciones. Asimismo, hay ciertas conexiones que tienen un costo, el cual se puede observar mencionado como +3000. En este mapase presenta una situación más realista en la que se considera más con conexiones entre estaciones así como más rutas. Como una situación muy realista resultaría en un grafo muy grande solo se ha considerado una pequeña parte de la ciudad:

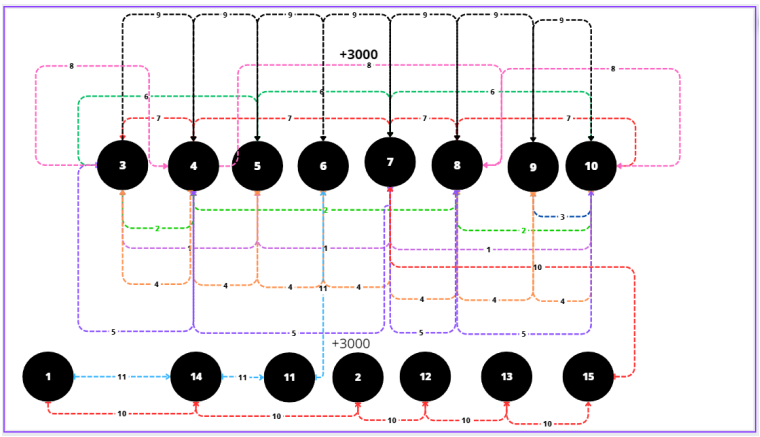

Fig. 11. Mapa escenario 1

1. Prueba de Priorización de minimizar transbordos En esta primera prueba con el escenario 2 se presenta la necesidad de desplazarse desde el nodo 1 hasta el nodo 10. Existen varias rutas para llegar, pero algunas requieren más de un transbordo. Que en su caso implicaria sacrificar un menor número de paradasm. Considerando que la importancia de minimizar el costo (0.1), los transbordos (0.8) y la cantidad de paradas (0.1), se muestra esta situación en la imagen adjunta (fig 12), donde se espera que únicamente se utilice la ruta 10 por lo tanto que se den 1 transbordos hasta al ruta 6,con un costo de 3 mil.

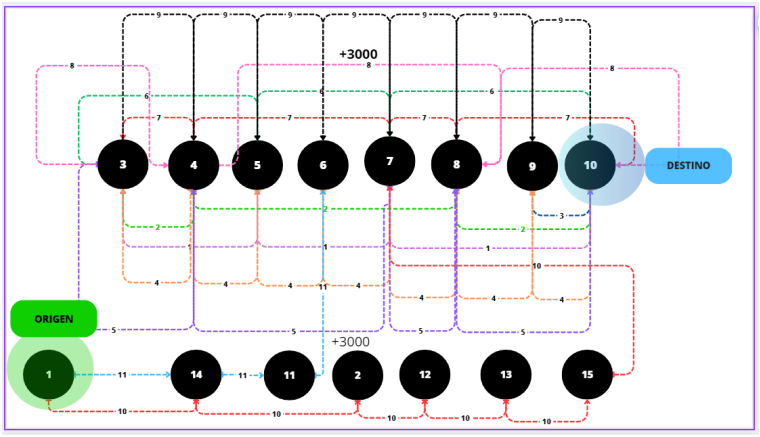

Fig. 12. Prueba 4:Resultados de minimizar las Transbordos y costos para llegar a la estación destino

Resultados del escenario 2 prueba 1: En efecto sigue una ruta es como se esperaba. Efectivamente se logró reducir los transbordos. A continuación se muestra la representación gráfica de la solución con (fig 13). Así mismo puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_2_esc1_1.py":

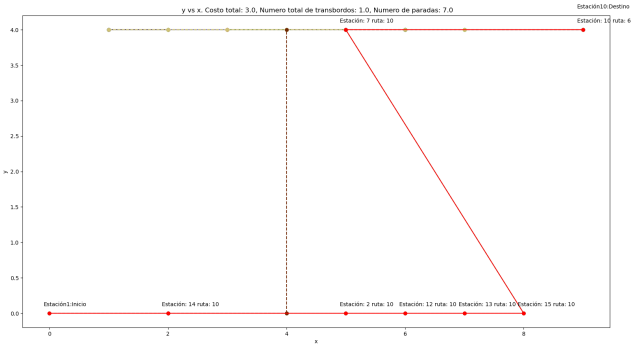

Fig. 13. Prueba 4:Resultados de minimizar las Transbordos y costos para llegar a la estación destino

Tenga en cuenta que el camino seleccionado está de color rojo

2. Priorización de minimizar Costos Considerando la importancia de minimizar el costo (0.8), los transbordos (0.1) y la cantidad de paradas (0.1) y considerando que el nodo origen es la estación 1 y el nodo destino es la estación 6. A continuación, se presenta el problema con la siguiente imagen (fig 14) se espera que únicamente se llegue a utilizar la ruta 10 y ruta 9.

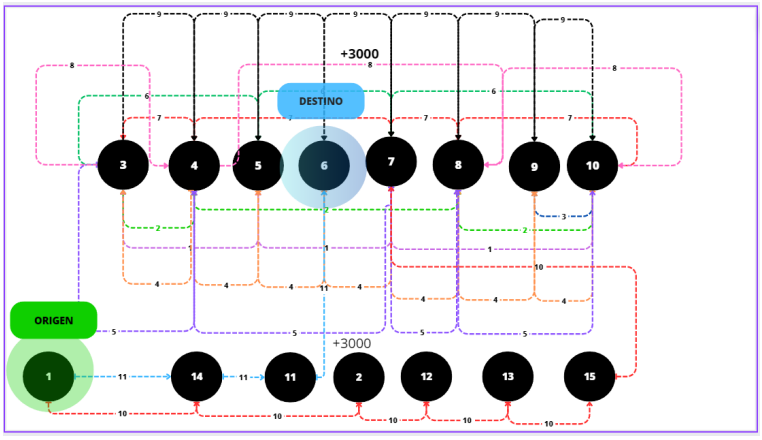

Fig. 14. Prueba 2: resultados de minimizar costos del viaje

Resultados del escenario 1 prueba 2: Se ha llegado a minimizar los costos correctamente, este se puede ver si es comparado al resultado de minimizar transbordos se ha llegado a solo tener que gastar 3 mil pesos. Sin embargo, se llevó a aumentar el número de paradas y transbordos a lo que hubiera pasado si se tomara la ruta 11 . En la siguiente grafica (fig 15) se muestra el trayecto a tomar. cabe resaltar que para ver la ejecución puede ejecutar el archivo .py con el nombre proyecto_2_esc2_2.py.

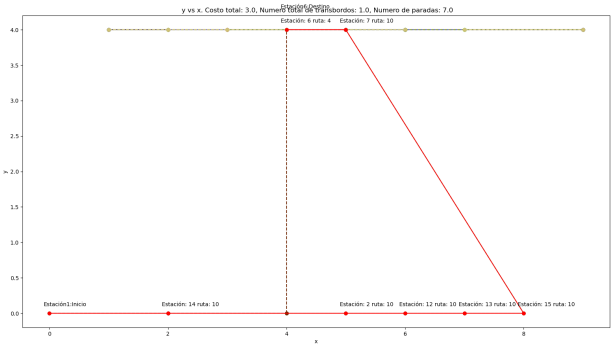

Fig. 15. Prueba 2: resultados de minimizar costos del viaje

Tenga en cuenta que el camino seleccionado está de color rojo

3. Priorización de minimizar paradas para llegar a la estación destino. Considerando el mismo mapa utilizado en el escenario uno, pero con un origen y destino diferentes, se busca minimizar el costo (0.1), los transbordos (0.1) y la cantidad de paradas (0.8). La situación se ilustra en la imagen adjunta, donde se espera utilizar exclusivamente la ruta 11, desde nodo 1 a nodo 6,. A continuación se puede ver una representación gráfica del problema planteado (fig 16):

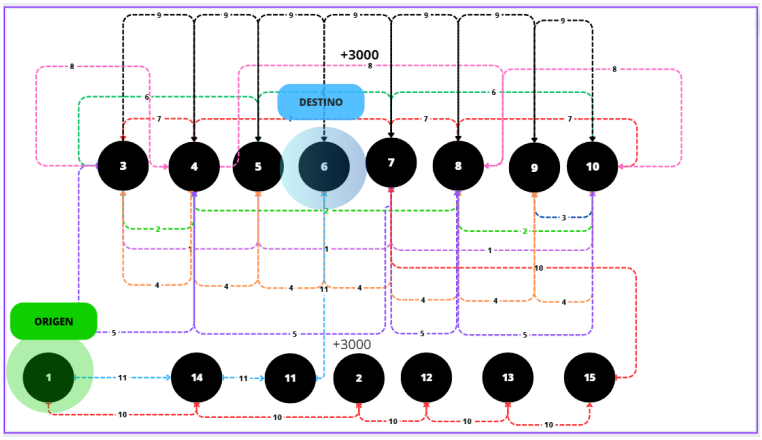

Fig. 16. Prueba 3: Planteamiento del problema de minimizar las paradas

Resultados del escenario 2 prueba 3: En efecto sigue una ruta similar a la esperada, logrando minimizar la cantidad de paradas, pero hay una elevación de costo esperada por usar la ruta 11, que la final solo implica 3 paradas. A continuación se puede ver una representación grafica de la solución. Así mismo este problema puede ejecutarlo con el archivo "proyecto_2_esc2_3.py".

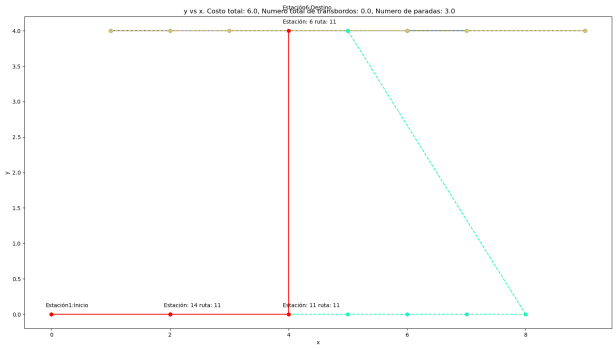

Fig. 17. Prueba 3:Resultados de minimizar las paradas

Tenga en cuenta que el camino seleccionado está de color rojo

4. Priorización de minimizar paradas y costos llegar a la estación destino. Considerando el mapa el siguiente mapa , se busca minimizar el costo (0.45), los paradas (0.45) y la cantidad de transbordos (0.1). La situación se ilustra en la imagen adjunta, puede llegar a ser complicada de decidir ya si se opta por la rutra 10 se espera tener que lidiar con muchas paradas y si se toma la ruta 11 se espera un costo elevado casi de 6.

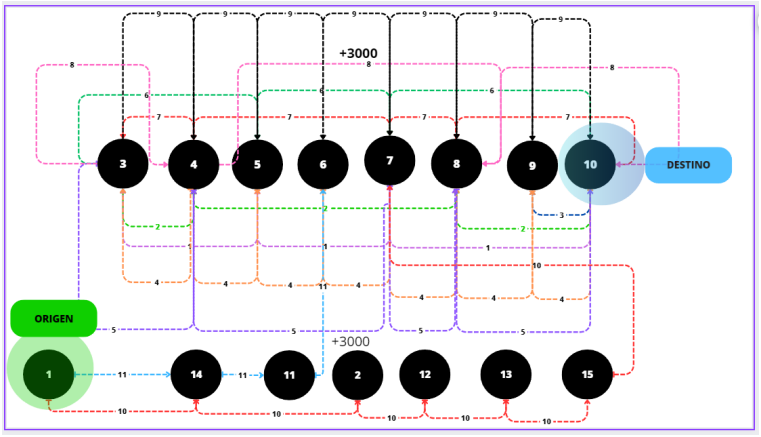

Fig. 18. Prueba 4: Planteamiento del problema de minimizar paradas y costos

Resultados del escenario 2 prueba 4: En efecto se sigue una ruta similar a la esperada, logrando minimizar la cantidad de paradas. Para llegar al nodo 4, siguió la vía más directa, con tal de minimizar costos se hizo un transbordos con la ruta 2 así evitando la conexión que hace la ruta 7 que tine costo de 3, asegurando solo 2 paradas en la ruta 3, en comparación con las 4 paradas de la ruta 4. Así mismo puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_2_esc1_4.py"

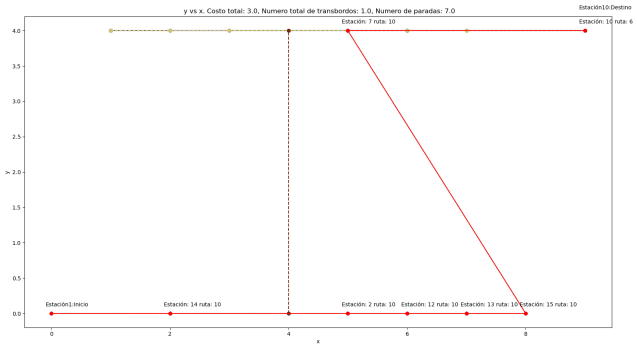

Fig. 19. Prueba 4:Resultados de minimizar las Transbordos y costos para llegar a la estación destino

5 Algoritmo propuesto

En el siguiente punto se proprone una heurística que ofrece una solución factible al problema, esta solución se basa en Dijkstra. Las diferencias se centran en la manera como se manejan los costos. Cabe resaltar que la solución no es completamente mía, se ha tomado en cuenta está solución propuesta por Max Reynolds en su articulo "Guide to Dijkstra's Algorithm in Python" [1].

5.1 Pseudo Algoritmo y explicación

Algorithm 1 Preprocesamiento costos de usar enlace.

1: $Initialize\ N = N$// número estaciones
2: $Initialize\ N = R$// número rutas
3: $Initialize\ inic = estacion\_inicial$// estación inicial
4: $Initialize\ end = estacion\_finall$// estación final
5: $Initialize\ alpha = \alpha$// Importancia de paradas
6: $Initialize\ beta = \beta$// Importancia de transbordos
7: $Initialize\ epsilon = \epsilon$// Importancia de costos
8: $Initialize\ data\_link = [S_{1,1,1}, S_{1,1,2}...S_{N,N,R}]$// i es nodo inicio, j nodo de llegada y k es la ruta
9: $Initialize\ data\_cost = [C_{1,1,1}, C_{1,1,2}...C_{N,N,R}]$// i es nodo inicio, j nodo de llegada y k es la ruta
10: $InitializeGraph\_linkedList = []$
//añade costos relacionados a importancia del enlace a él grafo
11: **for** $data\_link(i_1, j_1, k_1)$ *to* $data\_link(i_N, j_N, k_N)$ **do**
12: **if** $i \notin Graph\_linkedList$ **then**
13: $Graph\_linkedList[i] = []$
14: **end if**
15: **if** $j \notin Graph\_linkedList$ **then**
16: $Graph\_linkedList[j] = []$
17: **end if**
18: $Graph\_linkedList[i].append([j, k, \alpha])$ //añade costos relacionados a importancia del enlace a él grafo
19: **end for**

A diferencia del algoritmo pasado se ha optado por hacer una un grafo de costos que tenga calculado el costo de pasar por el enlace antes de aplicar el algorithmo de busqueda. Esto lo que quiere decir es que antes de aplicar djikstra se está primero sumando el calculo de pasar multiplicado por la importancia, de esta forma se contruye el grafo como una lista enlazada. En este fragmento de código se puede ver la generación del grafo con las lineas 11 a 19.

Algorithm 2 Preprocesamiento costos de usar enlace parte 2.

```
for data_cost(i_1, j_1, k_1) to data_cost(i_N, j_N, k_N) do
    lst_conecciones = Graph_linkedList[i]
    for x inst_conecciones.length() do
        if lst_conecciones[x][0] == j and lst_conecciones[x][1] == k then costo = data_cost(i, j, k)
            Graph_linkedList[i][x][2] = epsilon * costo + Graph_linkedList[i][x][2]
        end if
    end for
end for
```

En el algoritmo 2, se recorren los datos relacionados con los costos asociados al pasar por un enlace específico. Este proceso implica realizar un recorrido en el grafo para obtener la información relevante sobre el costo del enlace y actualizar su valor. De esta manera, se considera el costo actual mediante la suma del costo actual más epsilon (la importancia del costo), multiplicado por el costo proporcionado como parámetro.

Hasta este punto, hemos construido el grafo con los costos teniendo en cuenta la siguiente fórmula: $\epsilon * costo + \alpha * 1$, Dado que el tema de las rutas es relevante, pero no contamos con la información hasta después de aplicar el algoritmo, no es posible realizar un preprocesamiento completo que garantice el peso de los grafos.

Algorithm 3 Aplicación Djikstra.

```
1: Initialize distancia = [node_1 : Node(), node_2 : Node()...node_N : Node()]
2: Initialize visitando = [(0, inic, None)]
3: distancia[inic].d = 0
4: while(visitando.length! = 0) do
5: current_distance, current_node, current_route = visiting.pop()
6: if distancia[current_node].finished == false then
7:     distancia[current_node].finished = True
8:     for vecinos, ruta, costo ∈ Graph_linkedList[current_node] do
9:         if ruta! = current_route then
10:            cambio_ruta = beta
11:        end if
12:        if ruta == current_route then
13:            cambio_ruta = 0
14:        end if
15:        Costo_calculado = Graph_linkedList[current_node].d + costo + cambio_ruta
16:        if Costo_calculado < distancia[vecinos].d then
17:            distancia[vecinos].d = Costo_calculado
18:            distancia[vecinos].parent = current_node
19:            distancia[vecinos].route = ruta
20:        end if
21:    end for
22: end if
23: end while
```

En el algoritmo 3, se aplica Dijkstra, comenzando por la creación de dos listas, 'distancia' y 'visitando'. La primera lista indica el costo de llegar a un nodo específico desde el nodo inicial. Cabe destacar que 'Node' es una clase creada específicamente para manejar estos datos. La clase tiene 4 atributos (d: costo total, finished: ya fue visitado, route: la ruta tomada para llegar al nodo, parent: el nodo anterior antes de llegar). La segunda lista muestra una cola de prioridad organizada como un heap, que organiza los nodos a visitar.

El algoritmo comienza con el nodo inicial y un costo inicial de 0, junto con una ruta 'None' ya que aún no se ha tomado ninguna ruta. Al inicio, este nodo se elimina de la lista ya que se marcará como visitado, y se cambiará su estado a 'finalizado', ya que las líneas 8 a 21 llenarán la lista 'visitando' con los nodos vecinos.

En el bucle for (líneas 8 a 21), se recopila información sobre los nodos vecinos, incluyendo el nodo destino, la ruta y el peso. Gracias al preprocesamiento anterior, se tiene en cuenta el costo de usar el enlace previamente calculado. Dado que hasta este punto no se han considerado los transbordos (líneas 9 a 14), se verifica si hay un cambio de ruta. En caso de que haya un cambio de ruta, el valor de 'cambio de ruta' afectará al 'costo calculado' y, por lo tanto, será mayor.

Dentro de las líneas 16-20, se decide si el valor registrado en 'distancia' debe cambiar o no. Si el valor calculado cambia, también cambiarán los registros relacionados con el nodo anterior o la ruta. Este proceso se repite hasta que la lista 'visitando' quede completamente vacía, indicando que no hay más vecinos por visitar.

Algorithm 4 BackTracking para encontrar el camino y la ruta tomada

```
Initialize path = []
Initialize terminar = False
Initialize node = end
Initialize ruta = []
while(terminar == False) do
path.insert(0, distancias[node].parent)
ruta.insert(0, distancias[node].route)
node = distancias[node].parent
if node ==inic then
    terminar = True
end if
end while
```

En el algoritmo 4, se hace uso de las distancias calculadas en el algoritmo anterior. El propósito de este algoritmo es construir y devolver dos listas, path y ruta, que representan la ruta desde el nodo end hasta el nodo inic basándose en la información almacenada en el diccionario distancias. Para lograr esto, se construye la lista path obteniendo el padre (propiedad de Node) de cada nodo, y la lista ruta obteniendo la ruta (propiedad de Node) e insertándola al principio de la lista.

De esta manera, se logra obtener la ruta más óptima desde un nodo inicial hasta un nodo final. Por medio del archivo "proyecto_3_plantillaHeuristica.ipynb"puede evidenciar el proceso de plantear la heuristica

5.2 Resultados del Algoritmo vs Modelo matemático

Escenario 1

1. Prueba de Priorización de minimizar transbordos Para esta primera prueba se está usando el mismo mapa planteado en escenario 1. Se puede ver planteado el problema con la figura 3.

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	5	0.1	0.8	0.1

Table 3. Parametros Escenario 1: Prueba 1

Resultados del escenario 1 prueba 1: Heuristica En la imagen que se a continuación, puede dar evidencia de un resultado igual al que se ha llegado con el modelo matematico. Pues las rutas por las que pasa son las misma, la cantidad de transbordos es el mismo y el consto del viaje es el mismo. Solo se usa ruta 7 y se pasa por las paradas 1, 4 y 5. Puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_3_esc1_1.py"

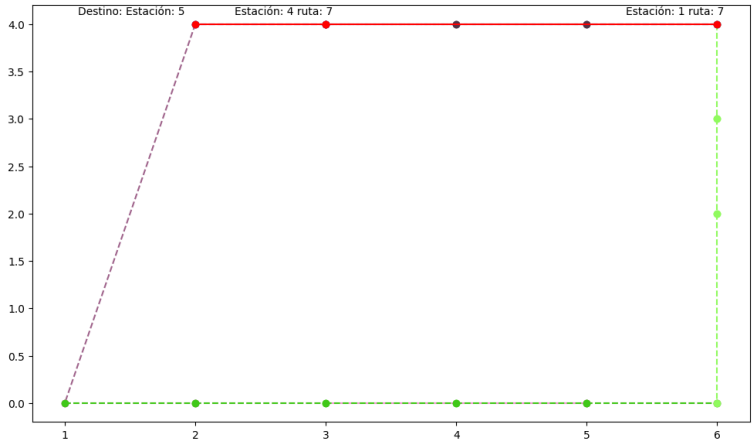

Fig. 20. Prueba 1:Resultados Priorizar minimizar transbordos

---

2. Priorización de minimizar Costos

Para esta segunda prueba se está usando el mismo mapa planteado en escenario 1. Se puede ver planteado el problema con la figura 3. En esta prueba se van a hacer uso de los siguientes parametros:

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	5	0.1	0.1	0.8

Table 4. Parametros Escenario 1: Prueba 2

Resultados del escenario 1 prueba 2: En la imagen que se a continuación, puede evidenciar que el resultado es igual al modelo matematico. Pues las rutas por las que pasa son las misma, la cantidad de transbordos es el mismo y el costo del viaje es el mismo. Solo se usa ruta 7 y 2, las cuales no tienen un costo adicionar porpasar y finalmente se pasa por las paradas 1, 4 y 5. Puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_3_esc1_2.py"

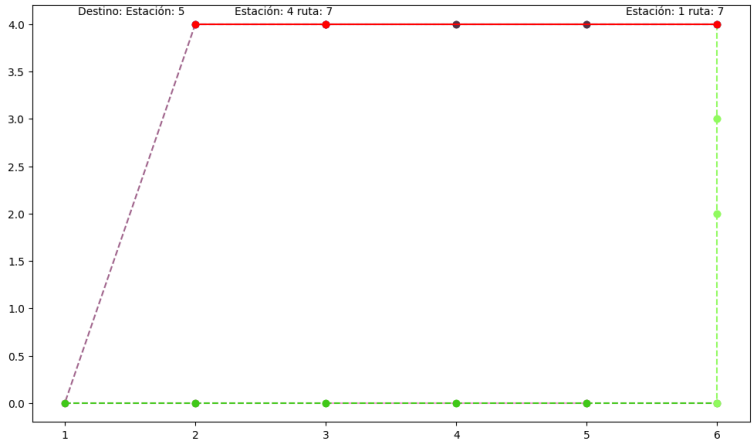

Fig. 21. Prueba 2 : Priorizar minimizar Costos

---

3. Priorización de minimizar paradas para llegar a la estación destino. Considerando el mismo mapa utilizado en el escenario uno, pero con un origen y destino diferentes. El problema está presentado en la figura 7. A continuación se puede ver la tabla con los parametros:

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	10	0.8	0.1	0.1

Table 5. Parametros Escenario 1: Prueba 3

Resultados del escenario 1 prueba 3: Se ha llegado al mismo resultado que se presenta en el módelo mátematico, con la misma ruta para llegar al destino. Se empieza con la ruta 7 hasta la estación 4, de la estación 4 se llega con la ruta 2 hasta la estación 5. En estación 5 se realiza un transbordo con ruta 3 para llegar a la estación 6, se hace una parada en la estación 7 y finalmente con la ruta 3 para llegar al destino en la estación 10.Puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_3_esc1_3.py"

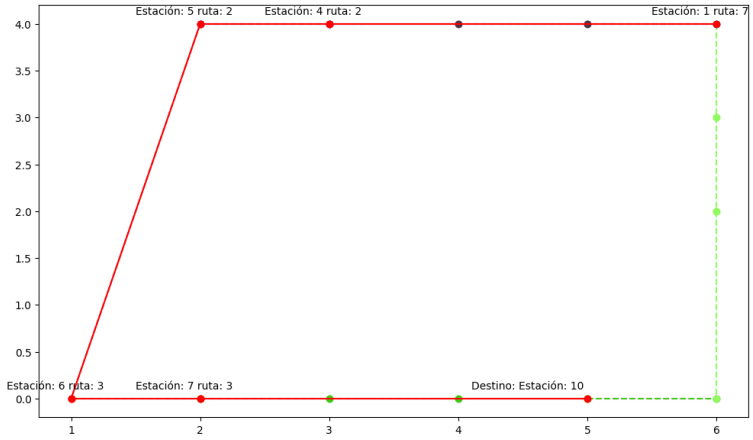

Fig. 22. Prueba 3:Resultados de minimizar las paradas

Tenga en cuenta que el camino seleccionado está de color rojo

4. Priorización de minimizar transbordos y costos llegar a la estación destino. Considerando el mismo problema anterior, representado con la figura 7. Y los siguientes parametros.

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	10	0.45	0.1	0.45

Table 6. Parametros Escenario 1: Prueba 4

Resultados del escenario 1 prueba 4: Se ha llegado al mismo resultado que en el modelo matematico, con las mismas rutas y mismas paradas y por lo tanto un mismo costo. Así mismo puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_3_esc1_4.py"

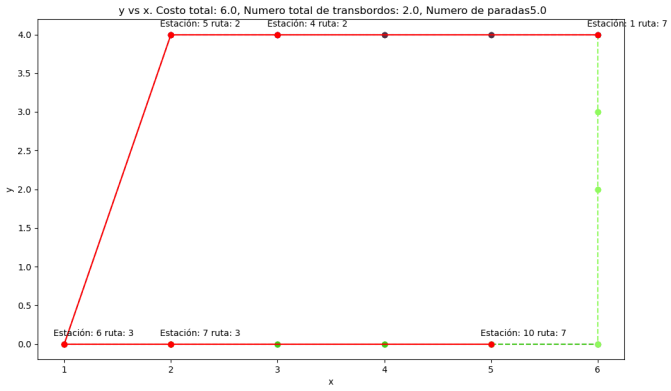

Fig. 23. Prueba 4:Resultados de minimizar las Transbordos y costos para llegar a la estación destino

Tenga en cuenta que el camino seleccionado está de color rojo

5.3 Escenario 2

1. Prueba de Priorización de minimizar transbordos En esta primera prueba con el escenario 2 se presenta la necesidad de desplazarse desde el nodo 1 hasta el nodo 10. La figura 12 ejemplifica en una grafica el problema. A continuación, se presenta los parametros para este problema:

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	10	0.1	0.8	0.1

Table 7. Parametros Escenario 2: Prueba 1

Resultados del escenario 2 prueba 1: Los resultados son iguales donde se comienza por la estación 1 , se sale con la ruta 10, que va a ir hasta la estación 7 y finalmente tomará la ruta 6 para llegar a la estación destino. Con el archivo "proyecto_3_esc2_1.py" se puede confirmar el correcto funcionamiento:

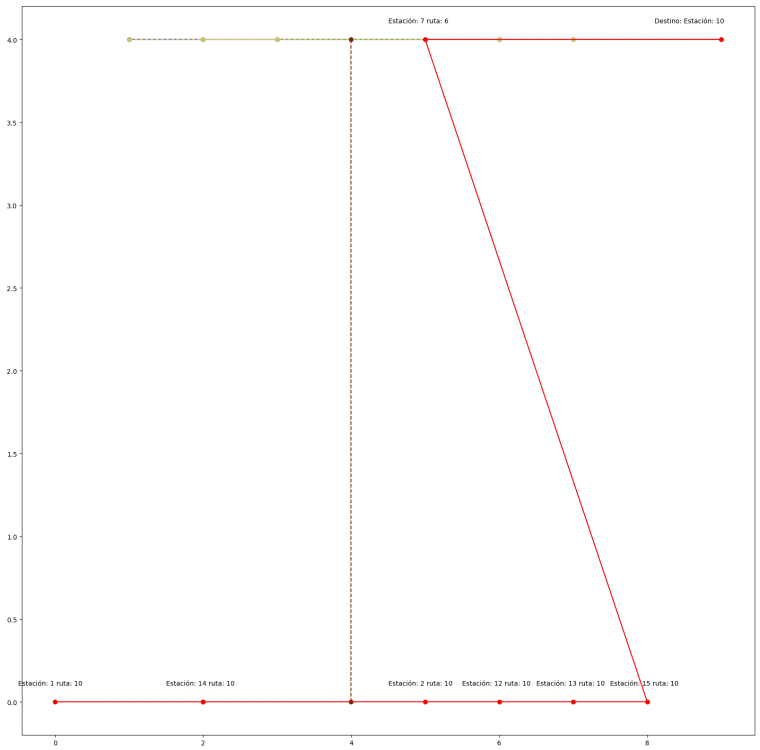

Fig. 24. Prueba 1:Resultados de Prueba de Priorización de minimizar transbordos

Tenga en cuenta que el camino seleccionado está de color rojo

2. Priorización de minimizar Costo Para esta segunda prueba se está usando el mismo mapa planteado en escenario 1. Se puede ver planteado el problema con la figura 14 . En esta prueba se van a hacer uso de los siguientes parametros:

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	6	0.1	0.1	0.8

Table 8. Parametros Escenario 1: Prueba 2

Resultados del escenario 2 prueba 2: En la imagen que se a continuación, puede evidenciar que el resultado es igual al modelo matematico. Pues las rutas por las que pasa son las misma, la cantidad de transbordos es el mismo y el costo

del viaje es el mismo. Solo se usa ruta 20 y 4. Puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_3_esc2_2.py"

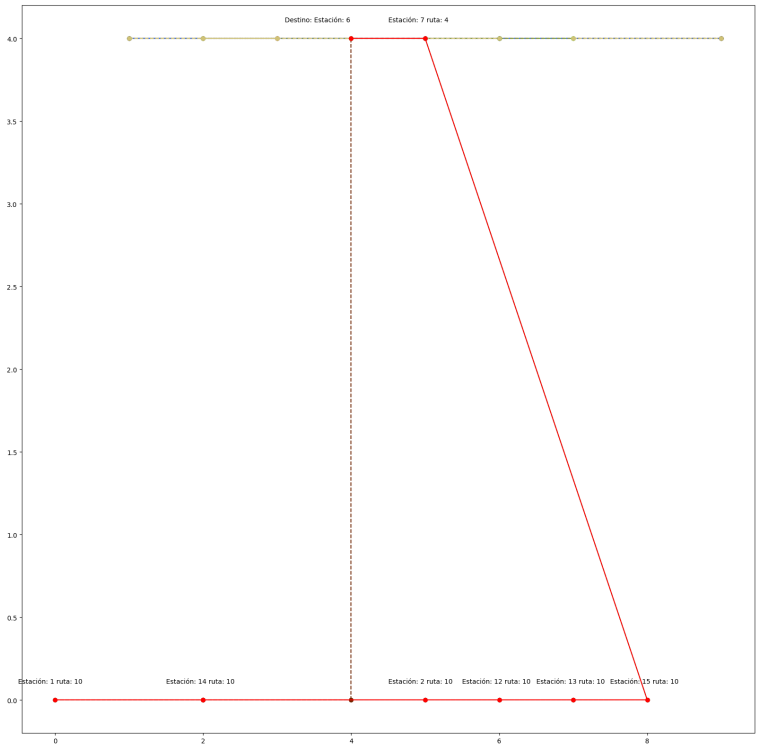

Fig. 25. Prueba 2 : Priorizar minimizar Costos

Tenga en cuenta que el camino seleccionado está de color rojo

3. Priorización de minimizar paradas para llegar a la estación destino. Considerando el mismo mapa utilizado en el escenario uno, pero con un origen y destino diferentes. El problema está presentado en la figura 16 . A continuación se puede ver la tabla con los parametros:

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	6	0.8	0.1	0.1

Table 9. Parametros Escenario 2: Prueba 3

Resultados del escenario 2 prueba 3: Se ha llegado al mismo resultado que se presenta en el módelo mátematico, con la misma ruta para llegar al destino. Solo se hace una ruta y se llega en 3 paradas.Puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_3_esc2_3.py"

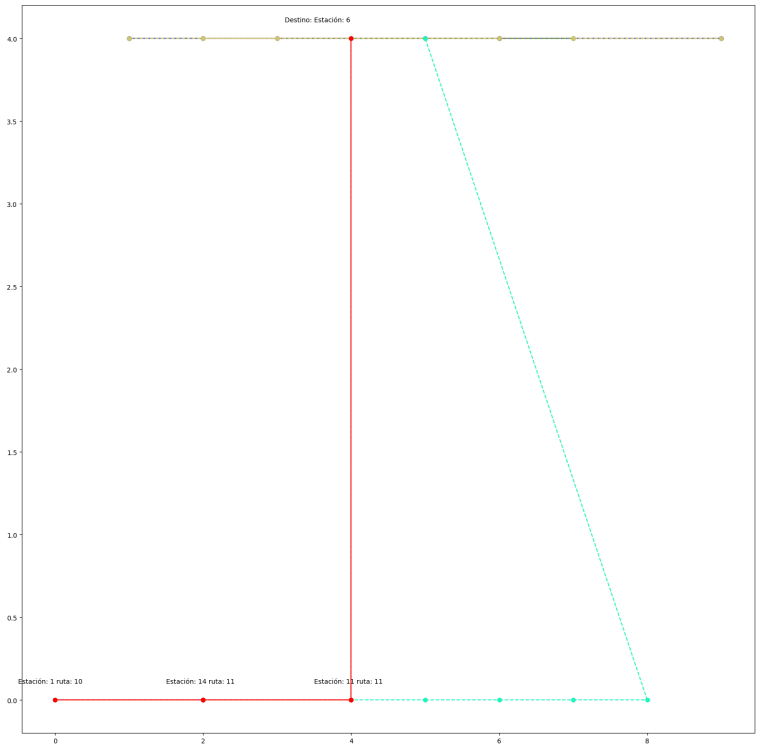

Fig. 26. Prueba 3:Resultados de minimizar las paradas

Tenga en cuenta que el camino seleccionado está de color rojo

4. Priorización de minimizar transbordos y costos llegar a la estación destino. Considerando el mismo problema anterior, representado con la figura 18. Y los siguientes parametros para decidir el camino a tomar.

Nodo Inicial	Nodo fin	α Importancia paradas	β Importancia transbordos	ϵ Importancia del costo
1	10	0.45	0.1	0.45

Table 10. Parámetros Escenario 1: Prueba 4

Resultados del escenario 2 prueba 4: Se ha llegado al mismo resultado que en el modelo matematico, con las mismas rutas y mismas paradas y por lo tanto un mismo costo. Así mismo puede confirma el funcionamiento del código con el archivo con el nombre "proyecto_3_esc2_4.py"

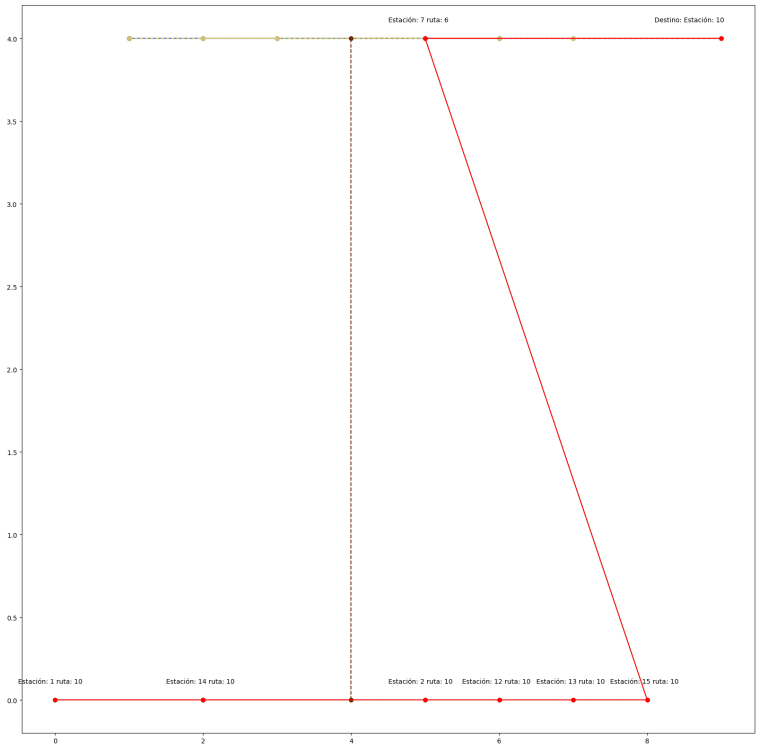

Fig. 27. Prueba 4:Resultados de minimizar las Transbordos y costos para llegar a la estación destino

Tenga en cuenta que el camino seleccionado está de color rojo

5.4 Conclusiones : Resultados Modelo Matemático vs Implementación Heuristica

En términos generales, los resultados obtenidos mediante la aplicación del Modelo Matemático no mostraron diferencias en comparación con los resultados de la Implementación Heurística. Aunque es necesario realizar un mayor número de pruebas para afirmar con certeza la efectividad de la heurística basada en el algoritmo de Dijkstra, hasta el momento se puede inferir que la aplicación de Dijkstra como heurística es una opción prometedora para abordar este problema en particular.

6 Bibliografía

[1]: Raynolds Max (2023), "Guide to Dijkstra's Algorithm in Python": BuiltIn. Recuperado de: https://builtin.com/software-engineering-perspectives/dijkstras-algorithm