Universidad Nacional Mayor de San Marcos

Decana de América. Universidad del Perú

Facultad de Ingeniería de Sistemas e Informática

Escuela Profesional de Ingeniería de Sistemas

APLICACIÓN DE RECORRIDOS Y ALGORITMOS DE GRAFOS EN LA REALIDAD – IDE CODE::BLOCKS C++

CURSO:

Estructura de Datos – Sección 3.

DOCENTE:

Salinas Azaña, Gilberto Anibal.

INTEGRANTES – GRUPO 10:

Dominguez Acosta, Prish Antony	**19200054**
Gutierrez Caruajulca, Javier Antonio	**19200020**
Malca Ramirez, Jhonattan David	**19200030**
Valerio Gamboa, Pedro Luis	**19200203**

LIMA – PERÚ

Miércoles 1 de septiembre del 2021

CONTENIDO TEMÁTICO

1. Introducción.

Actualmente existen muchos problemas en nuestro alrededor, muchos de ellos aún se encuentran en espera a ser resueltos, otros requieren de una optimización para mejorar sus grados de error y de esta forma obtener el mejor resultado posible.

En la práctica se puede observar que existen problemas o situaciones en que la información a almacenar no corresponde con un tipo de estructura. Para dar una solución a estos problemas se requiere de una estructura en la que se pueda representar otras relaciones entre los datos o componentes de la misma.

Se sabe que las gráficas cuentas con dos elementos, los cuales son: los vértices y las aristas que conectan un vértice con otro. Las aristas representan relaciones entre dicha información y los vértices almacenan información.

Estas estructuras tienen aplicaciones en diferentes campos, entre ellos se encuentra el transporte terrestre, marítimo, y aéreo, así como también en las redes de computadoras, mapas, ubicación geográfica, etc.

En el presente trabajo, se llevará a cabo una recopilación de información de diversos estudios ya realizados que estén relacionados al tema de estructura de datos de tipo árboles, con la finalidad de corroborar la solución ya establecida por los trabajos consultados a través de un programa desarrollado previamente. De estos trabajos, se obtendrá los datos necesarios para llevar a cabo una solución al problema con los algoritmos de Profundidad y Anchura, Dijkstra, Prim y Kruskal.

Se detallará los datos que se vayan a utilizar para resolver cada uno de los problemas a estudiar, así como también comparar los resultados obtenidos por el programa implementado y los resultados que muestren las fuentes consultadas.

2. Aplicaciones de grafos en problemas de la realidad.

2.1. Recorridos en profundidad y anchura.

Algoritmos alternos de bajo coste para la comparación de rutas metabólicas en plantas

1.1 Información proporcionada por el trabajo consultado.

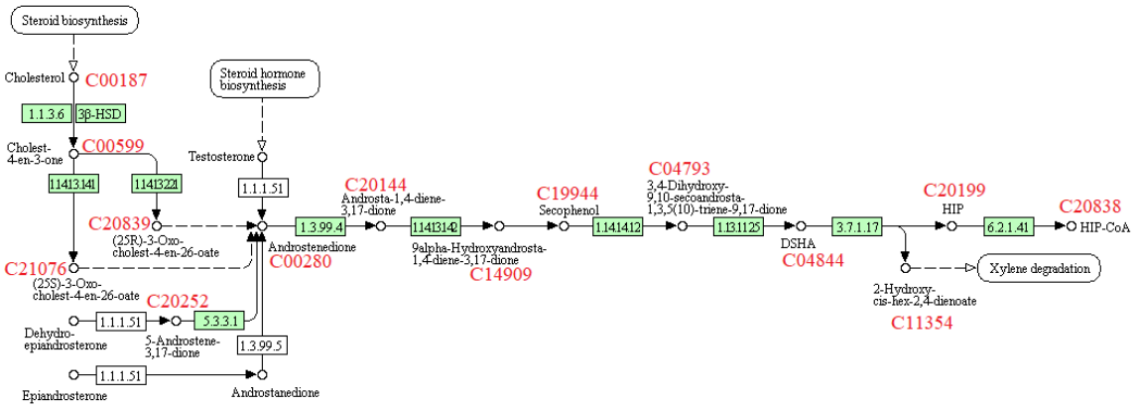
Las rutas metabólicas proveen información clave para alcanza un mejor entendimiento del ciclo de vida y sus procesos; información muy útil para que es usada para mejorar la medicina, la agronomía, farmacia y otras áreas similares.

Según (Meneses & Arias, 2017) mencionan que:

> "Las rutas metabólicas se modelan en computación como estructuras de datos tipo grafos. Dadas un par de rutas metabólicas, un alineamiento de dichas rutas corresponde a un mapeo entre subestructuras similares del par para proveer un valor de comparación o alineamiento".

Es por ello por lo que el uso de estructuras de datos tipo grafos es una herramienta de análisis que ayuda a tratar estas rutas metabólicas en base a la comparación entre las rutas.

Por ejemplo, tenemos la siguiente ruta metabólica "Steroid Biosynthesis"



Fuente 1. Ruta metabólica Steroid Biosynthesis, obtenido de

Se observa a los metabolitos representados como puntos blancos, enzimas como cajas y su número EC(Enzyme Commission numbers). Pero lo que nos interesa son las enzimas presentes en el organismo Steroid Biosynthesis, el cuál está representado por las cajas verdes.

1.2 Solución realizada con el programa implementado por el grupo

Para poder representarlos en un grafo utilizaremos el siguiente diccionario

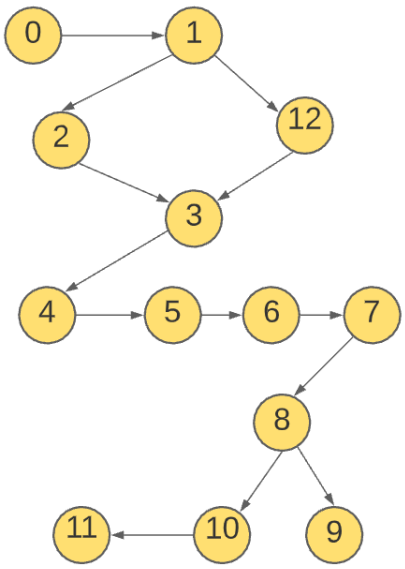
0 -> C00187	1-> C00599	2-> C20839
3 -> C00280	4 -> C20144	5 -> C14909

6 -> C19944 7 -> C04793 8 -> C04844

9 -> C11354 10 -> C20199 11 -> C20838

12 -> C21076

Con los cuales obtenemos el siguiente grafo



Fuente 2 Elaboración propia

Según su autor el recorrido por anchura de la ruta metabólica de Steroid Biosynthesis sería:

C00187 C00599 C21076 C20839 C00280 C20144 C14909 C19944 C04793 C04844 C20199 C11354 C20838

Lo que para nosotros se traduciría a:

0 1 12 2 3 4 5 6 7 8 10 9 11

En tanto, según su autor el recorrido por profundidad de la ruta metabólica de Steroid Biosynthesis sería el siguiente:

C00187 C00599 C20839 C00280 C20144 C14909 C19944 C04793 C04844 C11354 C20199 C20838 C21076

Lo que según nuestro grafo se traduciría a:

0 1 2 3 4 5 6 7 8 9 10 11 12

Lo cual podemos corroborar aplicando el algoritmo de recorrido en profundidad y anchura

1.2.1 Resultados obtenidos

Imagen 1. Elaboración propia

```
        Busqueda por profundidad
[0] [1] [12] [3] [4] [5] [6] [7] [8] [9] [10] [11]
[2]
8-9 10-11 8-10 7-8 6-7 5-6 4-5 3-4 12-3 1-12 0-1
        Tabla Busqueda Profundidad
   0    1    2    3    4    5    6    7    8    9    10    11    12    Vertice
   0    0    2    12    3    4    5    6    7    8    8    10    1    Padre de vertice
0.00 0.20 0.00 0.60 0.80 1.00 1.20 1.40 1.60 1.80 1.80 2.00 0.40    Peso del elemento entre vertice y su padre

        Busqueda por extension  [0]  [1]  [12]  [3]  [4]  [5]  [6]  [7]  [8]  [9]  [10]  [11]  [2]
```

Se muestra el resultado del algoritmo por profundidad en base a los datos ingresados del grafo de elaboración propia. Usando nuestro diccionario, podemos afirmar que el orden sería el siguiente:

C00187 C00599 C20839 C00280 C20144 C14909 C19944 C04793 C04844 C11354 C20199 C20838 C21076

Lo cual se llega a corroborar con el resultado planteado por el autor del documento que es parte de nuestro análisis.

2.- Aplicación en conexiones entre ciudades de un circuito turístico con distancias

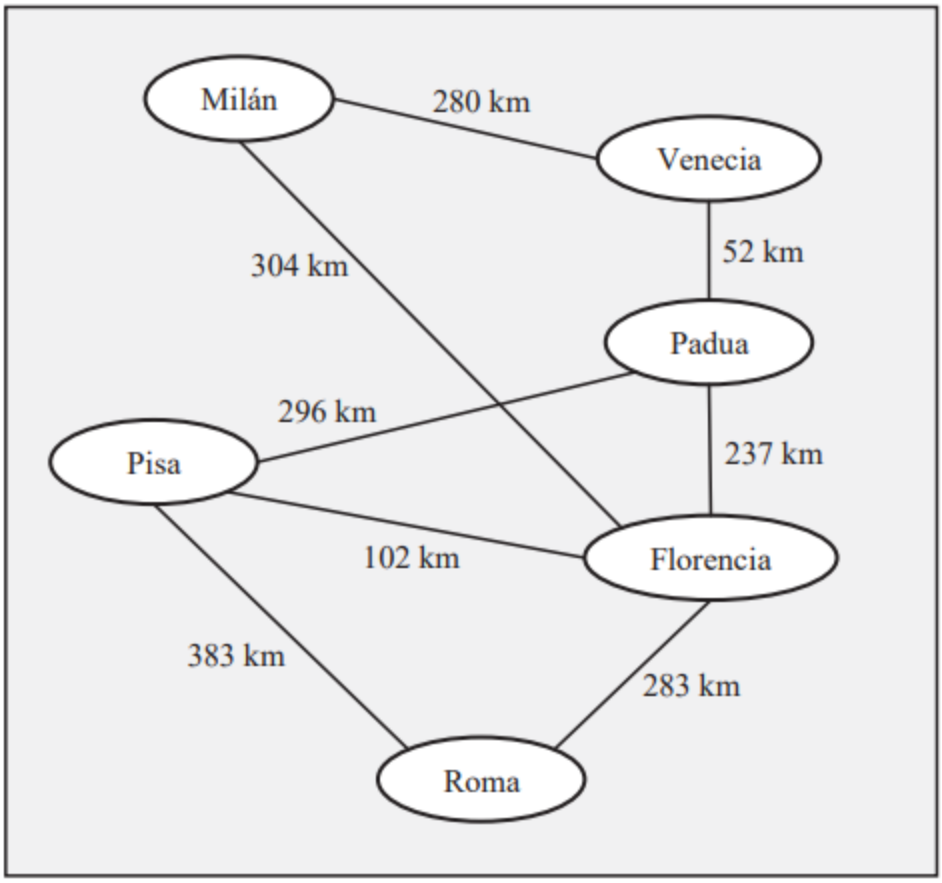
La aplicación de los algoritmos de recorridos en profundidad y anchura sobre trayectos, caminos o carreteras entre ciudades, distritos, calles y entre otros, se realiza mayormente para poder conocer el costo que nos tomaría en ir o visitar cada ciudad, distrito, calle y entre otros, siguiendo un patrón específico. Son problemas clásicos y de aplicación directa.

Según Fernández (2018) menciona que en un circuito turístico: "se trata de un problema en el que lo que se desea es recorrer todas las ciudades, los estados no estarán formados únicamente por la ciudad actual en la que se encuentra el viajero. Cada estado estará identificado por la lista ordenada de ciudades que se ha visitado al momento"

Es por ello la importancia al aplicar los algoritmos de recorridos por anchura o profundidad

2.1 Información proporcionada por el trabajo consultado

Figura 1. Grafo que representa las conexiones entre las ciudades que forman parte de un circuito turístico y sus distancias.



Fuente 3. Obtenido de Fernández (2018, p. 22)

Se observa los trayectos que existen entre las ciudades de Francia y la distancia que hay que recorrer por carretera si se desea ir de una ciudad a otra, también se observa que en algunos casos para poder llegar a un cierto destino se tendría que pasar obligatoriamente por una ciudad específica. Por ejemplo, para poder ir a Padua desde Milán, se tiene que pasar obligatoriamente por Venecia o Florencia.

1.2 Solución realizada con el programa implementado por el grupo

Para poder representar las conexiones entre las ciudades en un grafo que se acomode mejor al algoritmo del recorrido en profundidad y anchura. Utilizaremos el siguiente diccionario de elaboración propia:

Milán -> 0

Venecia -> 1

Padua -> 2

Pisa -> 3

Florencia -> 4

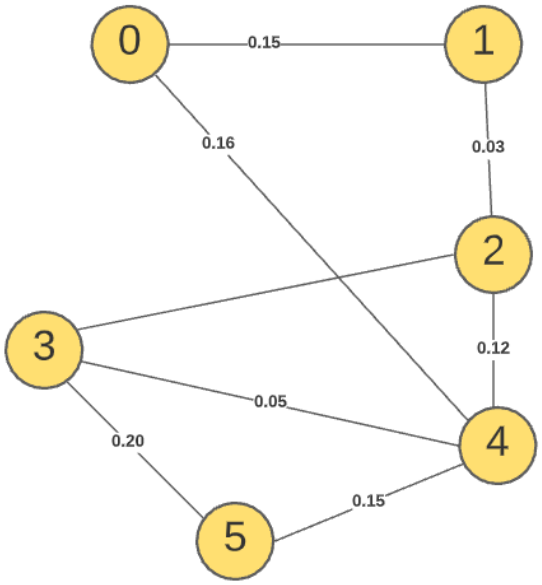
Roma -> 5

Adicionalmente utilizaremos la siguiente tabla, que nos ayudará a representar la distancia entre cada ciudad a través de un valor referencial, donde se trabaja con la frecuencia relativa en base al total de distancias (1937).

Distancia KM	Frecuencia relativa
280	0.15
304	0.16
52	0.03
296	0.15
237	0.12
102	0.05
383	0.20
283	0.15

Tabla de trayectos. Elaboración propia

Obteniendo así el siguiente grafo que viene a representar las conexiones entre las ciudades del circuito turístico planteado por el Fernández.

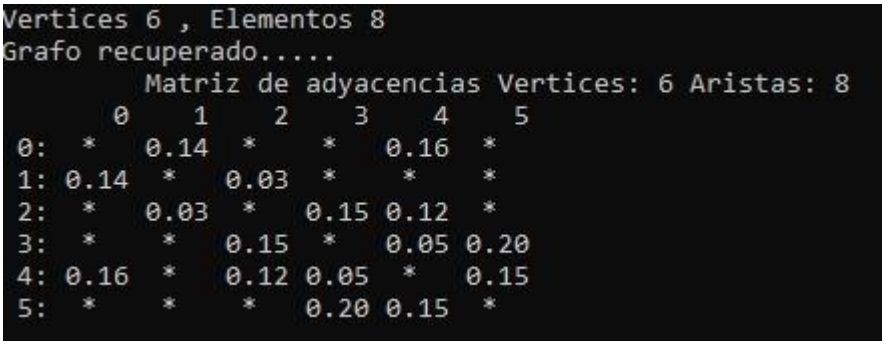


Fuente 4 Elaboración propia

1.2.1 Resultados obtenidos

Imagen 1. Elaboración propia

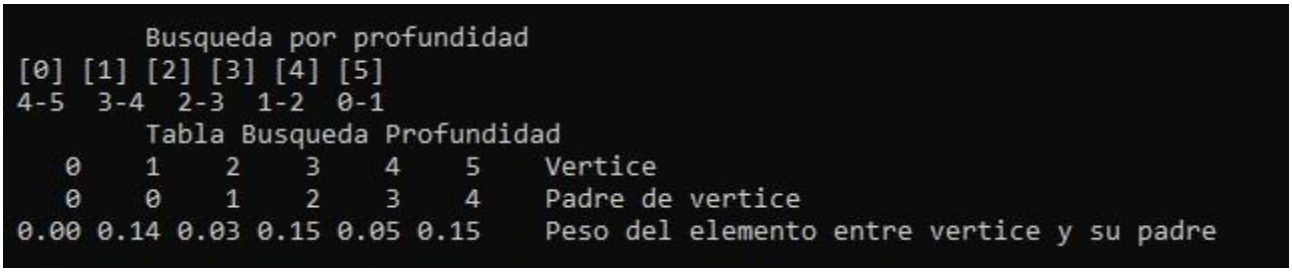
Matriz de adyacencias.



En la imagen 1 se muestra la matriz de adyacencias de acuerdo a los datos ingresados a través de un archivo de texto.

Imagen 2. Elaboración propia

Aplicación de búsqueda por profundidad



Se muestra el resultado del algoritmo por profundidad en base a los datos ingresados del grafo de elaboración propia. Usando nuestro diccionario, podemos afirmar que el orden sería el siguiente:

[Milán] [Venecia] [Padua] [Pisa] [Florencia][Roma]

Imagen 3. Elaboración propia

Recorrido en anchura

Se muestra el resultado del algoritmo por anchura en base a los datos ingresados del grafo. Usando nuestro diccionario podemos afirmar que el orden sería el siguiente:

[Milán] [Venecia] [Padua] [Pisa] [Florencia][Roma]

Tanto en el caso de del recorrido en anchura como en el de en profundidad se puede observar que los resultados dependen completamente de la situación de los nodos objetivos en el grafo, devolviendo siempre como resultado aquel nodo objetivo que se sitúe más arriba del árbol, en caso del recorrido en anchura, o más a la izquierda, en el caso de búsqueda en profundidad.

2.2. Algoritmo de Dijkstra.

2.2.1. Enrutamiento autónomo y gestión de energía de drones con acceso denegados a la navegación GPS a través del algoritmo Dijkstra.

Los espacios subterráneos han sido hasta el día de hoy uno de los mayores riesgos para el hombre, ya que es susceptible a riegos contra su seguridad y salud, por ejemplo, la presencia de gases peligroso, falta de ventilación, inestabilidad en las aberturas. Por lo tanto, es necesario el monitoreo de estos espacios con la finalidad de reducir el riesgo contra accidentes. El problema surge cuando se encuentra con espacios subterráneos que no son de fácil acceso, por ejemplo, rebajes abiertos, áreas abandonadas o con una alta concentración de contaminación, es entonces donde se hace necesario el uso del monitoreo remoto para la inspección de estos espacios.

Mirzaeinia, Shahmoradi, Roghanchi y Hassanalian (2019) manifiesta Existe muchas herramientas que hacen posible evaluar las condiciones de las áreas inaccesibles, entre todas ellas la que más destaca es el dron por su pequeño tamaño y maniobrabilidad.

Si bien es cierto en algunos drones vienen incorporados con un módulo GPS, no es posible su aplicación debido a que estos sistemas de navegación deben estar en áreas abiertas, existen además otras técnicas tales como radar, lídar y sonar, sin embargo, solo son apropiados para navegar por caminos rectos; como consecuencia los drones tienen un patrón de vuelo no optimizado que requerirá más energía que la necesaria. Por lo tanto, se hace necesario encontrar la ruta más corta de origen y destino.

Entonces lo que el autor propone es la creación de un nuevo sistema de navegación inteligente, usando el algoritmo de Dijkstra para el enrutamiento y gestión de energía de drones en entornos sin GPS.

2.2.1.1. Información proporcionada por el trabajo consultado.

Figura 1.

Modelado del metro subterráneo a investigar.

Fuente. Obtenido de Mirzaeinia, Shahmoradi, Roghanchi & Hassanalian (2019, p5)

El modelado muestra dos caminos diferentes para ir del punto A hacia el B, el primer camino 5 – 9 – 11 supone menos tiempo y por lo tanto, menos energía, en el caso el camino se encuentre congestionado, se realizará la búsqueda del próximo camino óptimo lo cual será posible gracias al uso de enrutadores en los escenarios.

Figura 2.

Representación en grafos del modelado de la figura 1.

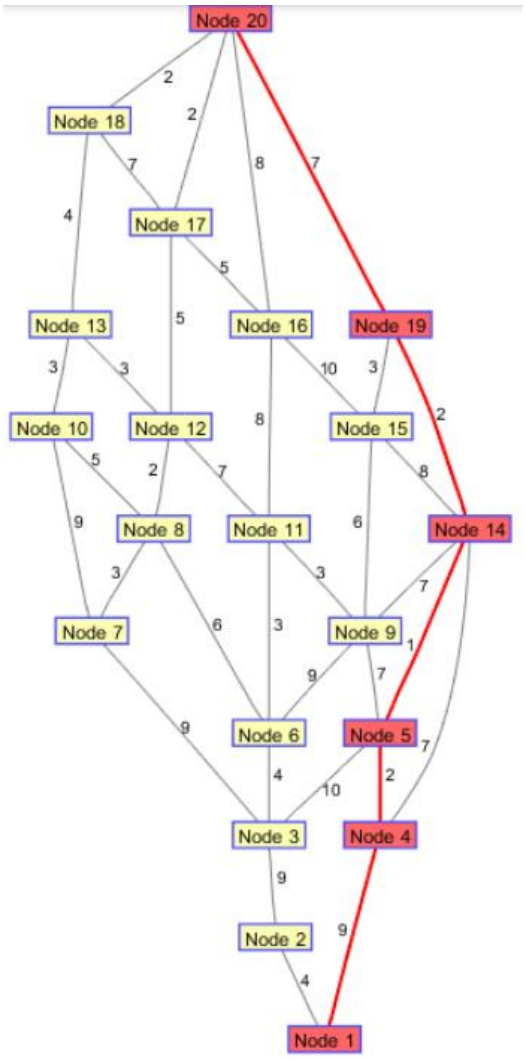


Figura 2

Representación en grafos del modelado de la figura 1

Fuente. Obtenido de Mirzaeinia, Shahmoradi, Roghanchi & Hassanalian (2019, p5)

El problema planteado por el autor es definir el camino mínimo de acuerdo con las estaciones (representado por nodos), en específico de la estación 1 a la 20; el algoritmo de enrutamiento entonces encontró la ruta que pasa por las estaciones número 0,3,4,13,18,19. Además se demostró que el peso total de la ruta es 21, siendo esta el camino mínimo para estas dos

estaciones, por lo tanto, este camino reducirá la potencia requerida de un dron para viajar a través de una red de metro.

1.2. Solución realizada con el programa implementado por el grupo.

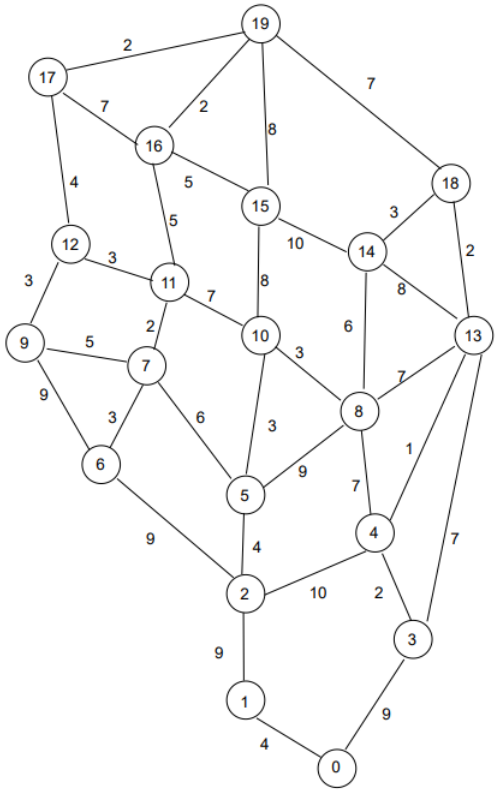
1.2. Tabla de pesos de acuerdo con las aristas del grafo del trabajo consultado.

Se ha trabajado con la frecuencia relativa en base al total de pesos (199), además para fines prácticos se ha cambiado el valor de los vértices del grafo, es decir al vértice 1 se cambió por el 0, vértice 2 por el 1, y así sucesivamente.

Vértice inicial	Vértice final	Peso	Frecuencia relativa
0	1	4	0.020101
0	3	9	0.045226
1	2	9	0.045226
2	6	9	0.045226
2	5	4	0.020101
2	4	10	0.050251
3	4	2	0.010050
3	13	7	0.035176
4	13	1	0.005025
4	8	7	0.035176
5	7	6	0.030151
5	8	9	0.045226
5	10	3	0.015075
6	7	3	0.015075
6	9	9	0.045226
7	9	5	0.025126
7	11	2	0.010050
8	10	3	0.015075
8	14	6	0.030151
8	13	7	0.035176
9	12	3	0.015075
10	11	7	0.035176
10	15	8	0.040201
11	12	3	0.015075
11	16	5	0.025126
12	17	4	0.020101
13	14	8	0.040201
13	18	2	0.010050

14	15	10	0.050251
14	18	3	0.015075
15	16	5	0.025126
15	19	8	0.040201
16	17	7	0.035176
16	19	2	0.010050
17	19	2	0.010050
18	19	7	0.035176

Obteniendo el siguiente grafo

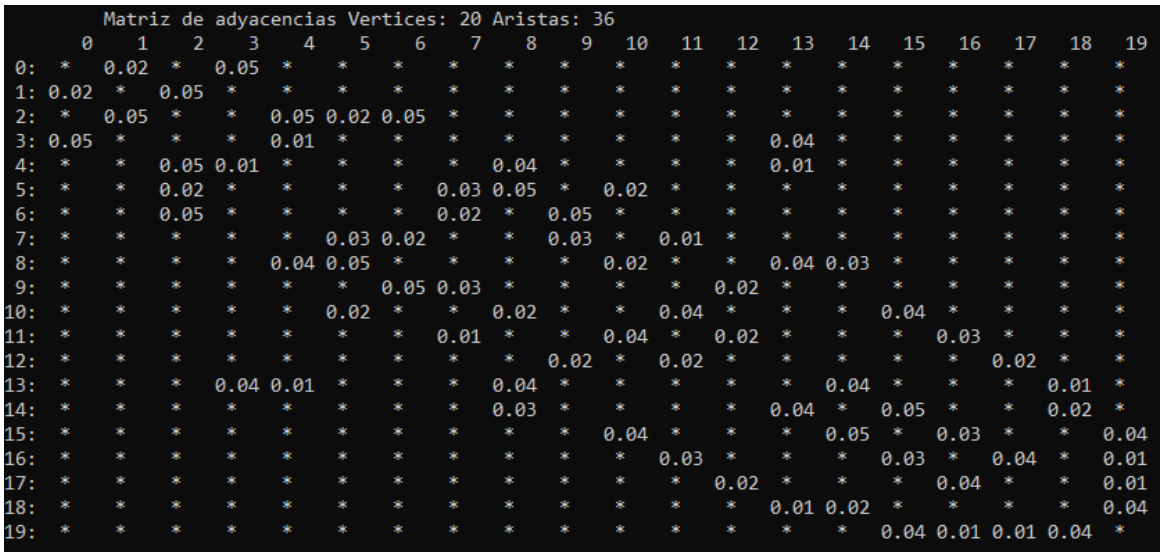


Fuente: Propia

1.2.1 Resultados obtenidos

Imagen 1

Matriz de adyacencias

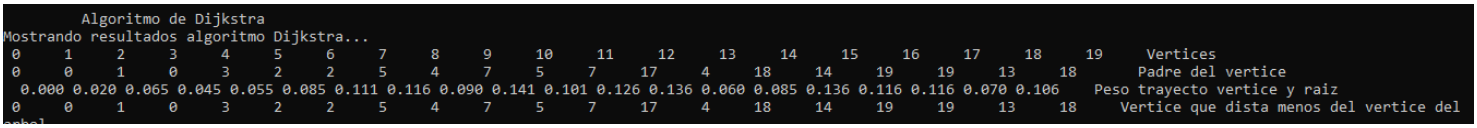


```
        Matriz de adyacencias Vertices: 20 Aristas: 36
      0    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19
 0:   *   0.02  *   0.05  *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *
 1: 0.02  *   0.05  *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *
 2:  *   0.05  *    *   0.05 0.02 0.05  *    *    *    *    *    *    *    *    *    *    *    *    *
 3: 0.05  *    *    *   0.01  *    *    *    *    *    *    *    *   0.04  *    *    *    *    *    *
 4:  *    *   0.05 0.01  *    *    *    *   0.04  *    *    *    *   0.01  *    *    *    *    *    *
 5:  *    *   0.02  *    *    *    *   0.03 0.05  *   0.02  *    *    *    *    *    *    *    *    *
 6:  *    *   0.05  *    *    *    *   0.02  *   0.05  *    *    *    *    *    *    *    *    *    *
 7:  *    *    *    *    *   0.03 0.02  *    *   0.03  *   0.01  *    *    *    *    *    *    *    *
 8:  *    *    *    *   0.04 0.05  *    *    *    *   0.02  *    *   0.04 0.03  *    *    *    *    *
 9:  *    *    *    *    *    *   0.05 0.03  *    *    *    *   0.02  *    *    *    *    *    *    *
10:  *    *    *    *    *   0.02  *    *   0.02  *    *   0.04  *    *    *   0.04  *    *    *    *
11:  *    *    *    *    *    *    *   0.01  *    *   0.04  *   0.02  *    *    *   0.03  *    *    *
12:  *    *    *    *    *    *    *    *    *   0.02  *   0.02  *    *    *    *    *   0.02  *    *
13:  *    *    *   0.04 0.01  *    *    *   0.04  *    *    *    *    *   0.04  *    *    *   0.01  *
14:  *    *    *    *    *    *    *    *   0.03  *    *    *    *   0.04  *   0.05  *    *   0.02  *
15:  *    *    *    *    *    *    *    *    *    *   0.04  *    *    *   0.05  *   0.03  *    *   0.04
16:  *    *    *    *    *    *    *    *    *    *    *   0.03  *    *    *   0.03  *   0.04  *   0.01
17:  *    *    *    *    *    *    *    *    *    *    *    *   0.02  *    *    *   0.04  *    *   0.01
18:  *    *    *    *    *    *    *    *    *    *    *    *    *   0.01 0.02  *    *    *    *   0.04
19:  *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.04 0.01 0.01 0.04  *
```

En la Imagen 1 se muestra la matriz de adyacencias de acuerdo con los datos ingresados en el archivo de texto, aparentemente los valores se repiten, pero no es así debido a que se han ingresado 6 cifras decimales para obtener un resultado más exacto. Es preciso señalar que, al mostrar los pesos de esta forma no se alterará los resultados obtenidos, fue considerado mostrarlo de esta manera para tener un mejor orden a la hora de mostrar los resultados.

Imagen 2

Aplicación del algoritmo de Dijkstra.



```
        Algoritmo de Dijkstra
Mostrando resultados algoritmo Dijkstra...
 0     1     2     3     4     5     6     7     8     9     10    11    12    13    14    15    16    17    18    19     Vertices
 0     0     1     0     3     2     2     5     4     7     5     7     17    4     18    14    19    19    13    18       Padre del vertice
  0.000 0.020 0.065 0.045 0.055 0.085 0.111 0.116 0.090 0.141 0.101 0.126 0.136 0.060 0.085 0.136 0.116 0.116 0.070 0.106    Peso trayecto vertice y raiz
 0     0     1     0     3     2     2     5     4     7     5     7     17    4     18    14    19    19    13    18     Vertice que dista menos del vertice del
arbol
```

Se muestra el resultado del algoritmo de Dijkstra en base a los datos ingresados. Se obtuvo el siguiente resultado en base al mínimo recorrido desde de 19 hacia el 0

[19] – [18] – [13] – [4] – [3] – [0]

Los cuales concuerdan con lo señalado por el autor, en lo referente a pesos se obtuvo lo siguiente:

vértice inicial	vértice final	Peso	Frecuencia relativa
18	19	7	0.03517588
13	18	2	0.01005025
4	13	1	0.00502513
3	4	2	0.01005025
0	3	9	0.04522613

Por lo tanto, se puede concluir que el programa implementado ha sido capaz de obtener el resultado y dar solución al problema del mínimo recorrido

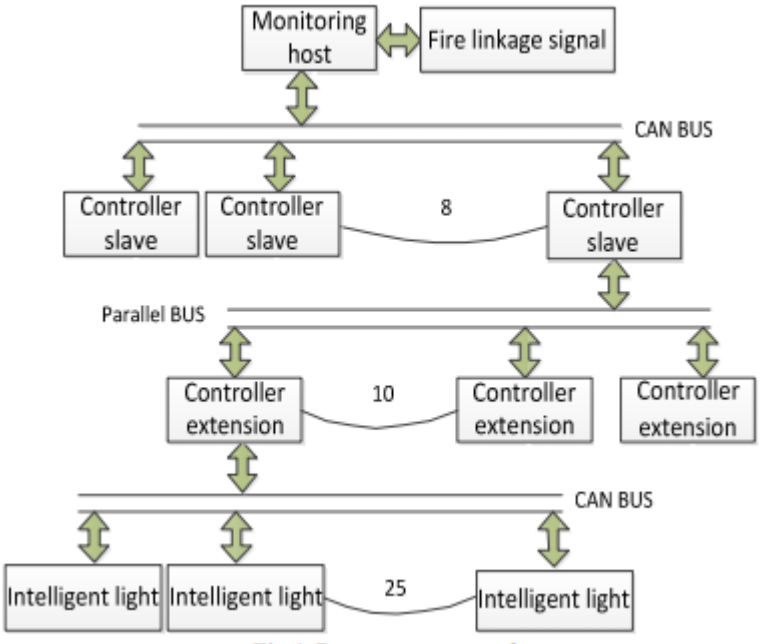
1. **Aplicación del Algoritmo de Dijkstra en un sistema inteligente de evacuación ante incendios**

El país de China, el cual concentra la mayor parte de habitantes de la tierra, se ha visto en la necesidad de construir complejos y grandes edificios. El problema radica en que el sistema tradicional de evacuación de incendios no puede determinar la situación actual de incendio y lo que es más importante, no puede controlar la dirección de las luces de evacuación de emergencia, las cuales a menudo confunden a las personas.

Frente a este problema el sistema planteado como solución indica una dirección segura para escapar del incendio a través de luces de evacuación de emergencia para que la gente pueda evitar el humo, calor colectivo y las llamas de manera efectiva a través del uso del algoritmo de Dijkstra.

Xu, Wang, Zheng y Hang explican que el sistema se divide en cuatro niveles de estructura, que son el host de monitoreo, el controlador esclavo, la extensión del controlador y los dispositivos terminales inteligentes

Figura 3

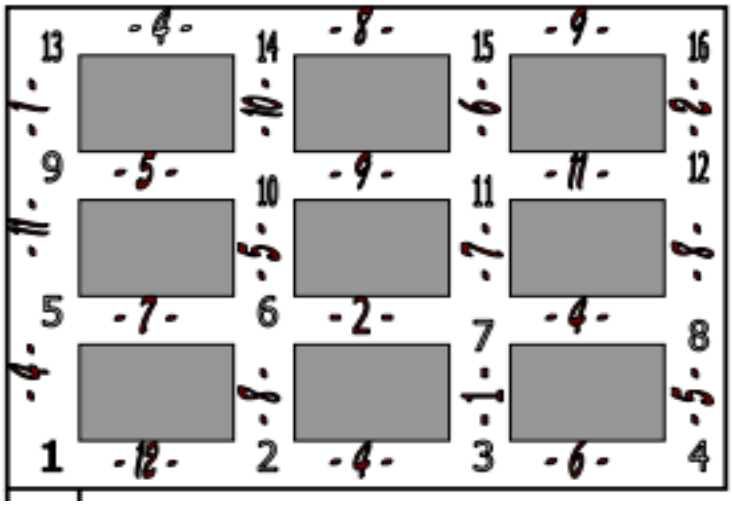


Fuente: Obtenido de Xu, Wang, Zheng y Hang (2012, p1)

El modelo requiere tener en cuenta los factores relevantes que influyen en la evacuación de un incendio tañes como el entorno del lugar, el espacio, velocidad de propagación del humo, etc.

Entonces los pesos de cada camino estarán en función a distintos factores tales como: velocidad de propagación del humo, gases tóxicos de aire, calor convectivo en la carretera para escapar, la velocidad de escape, la densidad de población, el estado de congestión del camino se puede concluir que mientras el peso del camino sea mayor es porque las condiciones serán las peores. Para la aplicación del sistema el autor plantea un plano de un centro comercial que se muestra en la figura:

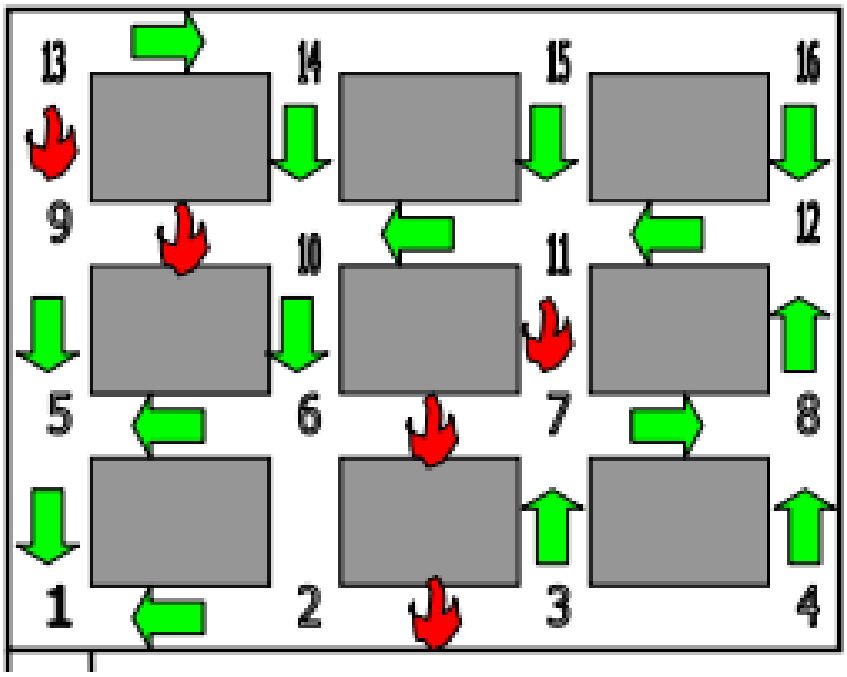
Figura 4



Fuente: Obtenido de Xu, Wang, Zheng y Hang (2012, p3)

Siendo 1 la salida del establecimiento, además, en caso el camino de un vértice a otro sea imposible de pasar, debido al fuego el peso se anula es decir ya no existirá una conexión entre los vértices, por lo tanto, el sistema debe ser capaz de iluminar el camino que sea el de menor peso posible. Esto puede evidenciarse en la figura 5

Figura 5



Fuente: Obtenido de Xu, Wang, Zheng y Hang (2012, p3)

Luego obtiene la siguiente tabla con los resultados de aplicar el algoritmo.

Node	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Shorest distance	12	49	49	4	11	48	44	15	16	25	36	30	26	31	38
The corresponding path to source point	1 2	1 5 6 10 11 12 8 7 3	1 5 6 10 11 12 8 4	1 5	1 5 6	1 5 6 10 11 12 8 7	1 5 6 10 11 12 8	1 5 9	1 5 6 10	1 5 6 10 11	1 5 6 10 11 12	1 5 6 10 14 13	1 5 6 10 14	1 5 6 10 11 15	1 5 6 10 11 12 16

1.2. Solución realizada con el programa implementado por el grupo.

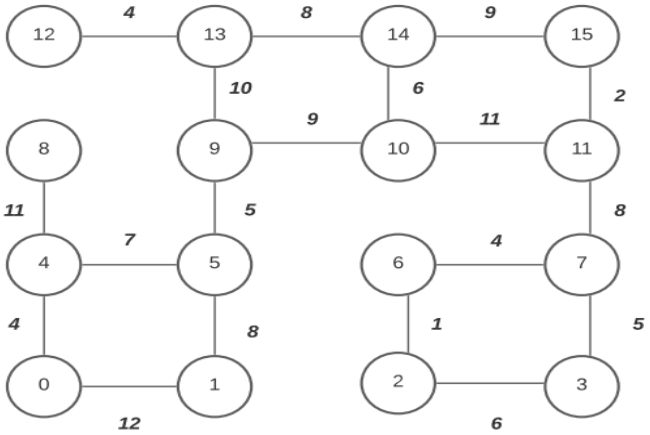
1.2. Tabla de pesos de acuerdo con las aristas del grafo del trabajo consultado.

Se ha trabajado con la frecuencia relativa en base al total de pesos (138), además para fines prácticos se ha cambiado el valor de los vértices del grafo, es decir al vértice 1 se cambió por el 0, vértice 2 por el 1, y así sucesivamente.

vértice inicial	vértice final	Peso	Frecuencia relativa
0	1	12	0.08696
0	4	4	0.02899
1	5	8	0.05797
2	3	6	0.04348
2	6	1	0.00725
3	7	5	0.03623
4	8	11	0.07971
4	5	7	0.05072
5	9	5	0.03623
6	7	4	0.02899
7	11	8	0.05797
9	13	10	0.07246
9	10	9	0.06522
10	14	6	0.04348
10	11	11	0.07971
11	15	2	0.01449
11	7	8	0.05797
12	13	4	0.02899
13	14	8	0.05797
14	15	9	0.06522

Obtuvimos el siguiente grafo

Figura 6



Fuente: Propia

1.2.1 Resultados obtenidos

Imagen 1

```
Vertices 16 , Elementos 19
Grafo recuperado.....
        Matriz de adyacencias Vertices: 16 Aristas: 19
      0    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
 0:   *   0.09  *    *   0.03  *    *    *    *    *    *    *    *    *    *    *
 1: 0.09  *    *    *    *   0.06  *    *    *    *    *    *    *    *    *    *
 2:   *    *    *   0.04  *    *   0.01  *    *    *    *    *    *    *    *    *
 3:   *    *   0.04  *    *    *    *   0.04  *    *    *    *    *    *    *    *
 4: 0.03  *    *    *    *   0.05  *    *   0.08  *    *    *    *    *    *    *
 5:   *   0.06  *    *   0.05  *    *    *    *   0.04  *    *    *    *    *    *
 6:   *    *   0.01  *    *    *    *   0.03  *    *    *    *    *    *    *    *
 7:   *    *    *   0.04  *    *   0.03  *    *    *    *   0.06  *    *    *    *
 8:   *    *    *    *   0.08  *    *    *    *    *    *    *    *    *    *    *
 9:   *    *    *    *    *   0.04  *    *    *    *   0.07  *    *   0.07  *    *
10:   *    *    *    *    *    *    *    *    *   0.07  *   0.08  *    *   0.04  *
11:   *    *    *    *    *    *    *   0.06  *    *   0.08  *    *    *    *   0.01
12:   *    *    *    *    *    *    *    *    *    *    *    *    *   0.03  *    *
13:   *    *    *    *    *    *    *    *    *   0.07  *    *   0.03  *   0.06  *
14:   *    *    *    *    *    *    *    *    *    *   0.04  *    *   0.06  *   0.07
15:   *    *    *    *    *    *    *    *    *    *    *   0.01  *    *   0.07  *
```

En la Imagen 1 se muestra la matriz de adyacencias de acuerdo con los datos ingresados en el archivo de texto, aparentemente los valores se repiten, pero no es así debido a que se han ingresado 6 cifras decimales para obtener un resultado más exacto. Es preciso señalar que, al mostrar los pesos de esta forma no se alterará los resultados obtenidos, fue considerado mostrarlo de esta manera para tener un mejor orden a la hora de mostrar los resultados.

Imagen 2

Aplicación del algoritmo de Dijkstra.

```
        Algoritmo de Dijkstra
Mostrando resultados algoritmo Dijkstra...
 0     1     2     3     4     5     6     7     8     9     10    11    12    13    14    15     Vertices
 0     0     6     7     0     4     7     11    4     5     9     10    13    9     10    11      Padre del vertice
0.000 0.087 0.355 0.355 0.029 0.080 0.348 0.319 0.109 0.116 0.181 0.261 0.217 0.188 0.225 0.275    Peso trayecto vertice y raiz
 0     0     6     7     0     4     7     11    4     5     9     10    13    9     10    11     Vertice que dista menos del vertice del arbol
```

Se muestra el resultado del algoritmo de Dijkstra en base a los datos ingresados. Se obtuvo el siguiente resultado en base al mínimo de todos los vértices para poder llegar al vértice 0, como ejemplo tomaremos al vértice de partida [2] lo cual genera el siguiente camino:

[2] – [6] – [7] – [11] – [10] – [9] - [5] – [4] – [0]

Los cuales concuerdan con lo señalado por el autor, en lo referente a pesos se obtuvo lo siguiente:

vértice	Frecuencia relativa	Peso
1	0.087	12
2	0.355	49
3	0.355	49
4	0.029	4
5	0.08	11
6	0.348	48
7	0.319	44
8	0.109	15
9	0.116	16
10	0.181	25
11	0.261	36
12	0.217	30
13	0.188	26
14	0.225	31
15	0.275	38

Por lo tanto, se puede concluir que el programa implementado ha sido capaz de obtener el resultado y dar solución al problema del mínimo recorrido.

2.3. Algoritmo de Prim.

2.3.1. Optimización de la planificación de la trayectoria de la fibra óptica.

Hoy en día es muy notoria la transición del cable de cobre a la fibra óptica, este cambio impulsa el desarrollo de una tecnología que puede transmitir los datos con mayor rapidez y precisión. Este cable de fibra óptica es muy costoso, por dicho motivo es fundamental realizar una óptima instalación.

Iqball, Utama, Purba y Purwanto (2017) expresan que determinar la cantidad de cable que se utiliza en el momento de la instalación es difícil si se hace manualmente. El

algoritmo de Prim puede optimizar calculando el árbol de expansión mínimo en las ramas utilizadas para la instalación del cable de fibra óptica. Este algoritmo puede utilizarse para acortar el tiempo hasta un destino haciendo que todos los puntos estén interconectados según los puntos enumerados.

La construcción de una red de fibra óptica es un gran reto, puesto que hay muchos obstáculos al querer realizar esta tarea, teniendo como obstáculo principal el alto costo del cable. Un mal cálculo en la construcción puede perjudicar a ciertas zonas del proyecto, es por ello que se debe de buscar el algoritmo adecuado que ayude a encontrar la ruta más corta y por ende a reducir costos.

Iqball et.al (2017) concluyen que el Algoritmo de Prim es un algoritmo que puede utilizarse para calcular el uso mínimo de claves para minimizar el coste. Este algoritmo buscara conexiones más cortas entre los dos puntos de ramificación del cable.

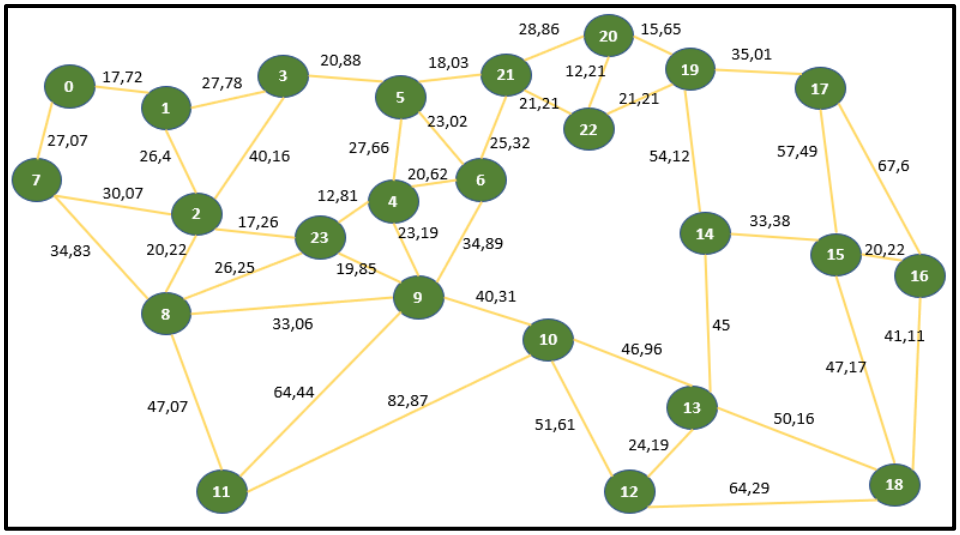
Los autores del trabajo realizada esperan que este uso reduzca el número de cables utilizados en cada poste transmisor-

2.3.1.1. Información proporcionada por el trabajo consultado.

2.3.1.1.1. Grafo de la ruta del cable de fibra óptica.

Figura 1.

Ruta gráfica de la fibra óptica.



Nota. Adaptado de Fiber optic graph route. Tomado de Prim´s Algorithm for Optimizing Fiber Optic Trajectory Planning (p.506), por Iqball, Utama, Purba & Purwanto, 2017, International Journal of Scientific Research in Science and Technology.

La figura 1 es un gráfico que fue formado a partir de coordenadas, donde cada nodo está conectado directamente a sus ramas y cada intersección de nodos tiene pesos. Cada vértice representa un poste transmisor y cada arista una longitud de cable utilizado.

2.3.1.1.2. Tabla de pesos de acuerdo con las coordenadas.

Tabla 1.

Tabla de Pesos.

Nodo 1	Nodo 2	Peso
[0]	[1]	17,72
[0]	[7]	27,07
[1]	[0]	17,72
[1]	[2]	26,4
[1]	[3]	27,78
[2]	[1]	26,4
[2]	[3]	40,16
[2]	[7]	30,07
[2]	[8]	20,22
[2]	[23]	17,26
[3]	[1]	27,78
[3]	[2]	40,16
[3]	[5]	20,88
[4]	[5]	27,66
[4]	[6]	20,62
[4]	[9]	23,19
[4]	[23]	12,81
[5]	[4]	27,66
[5]	[6]	23,02
[5]	[21]	18,03
[6]	[4]	20,62
[6]	[5]	23,02
[6]	[9]	34,89
[6]	[21]	25,32
[7]	[0]	27,07
[7]	[2]	30,07
[7]	[8]	34,83
[8]	[2]	20,22
[8]	[7]	34,83
[8]	[9]	33,06
[8]	[11]	47,07
[8]	[23]	26,25
[9]	[4]	23,19
[9]	[6]	34,89

[9]	[8]	33,06
[9]	[10]	40,31
[9]	[11]	64,44
[9]	[23]	19,85
[10]	[9]	40,31
[10]	[11]	82,87
[10]	[12]	51,61
[10]	[13]	46,96
[11]	[8]	47,07
[11]	[9]	64,44
[11]	[10]	82,87
[12]	[10]	51,61
[12]	[13]	24,19
[12]	[18]	64,29
[13]	[10]	46,96
[13]	[14]	45
[13]	[18]	50,16
[14]	[13]	45
[14]	[15]	33,38
[14]	[19]	54,12
[15]	[14]	33,38
[15]	[16]	20,22
[15]	[17]	57,49
[15]	[18]	47,17

[16]	[15]	20,22
[16]	[17]	67,6
[16]	[18]	41,11
[17]	[15]	57,49
[17]	[16]	67,6
[17]	[19]	35,01
[18]	[12]	64,29
[18]	[13]	50,16
[18]	[15]	47,17
[18]	[16]	41,11
[19]	[14]	54,12
[19]	[17]	35,01
[19]	[20]	15,65
[19]	[22]	21,21
[20]	[19]	15,65
[20]	[21]	28,86
[20]	[22]	12,21
[21]	[5]	18,03
[21]	[6]	25,32
[21]	[20]	28,86
[21]	[22]	21,21
[22]	[19]	21,21
[22]	[21]	21,21

[23]	[2]	17,26
[23]	[4]	12,81
[23]	[8]	26,25
[23]	[9]	19,85

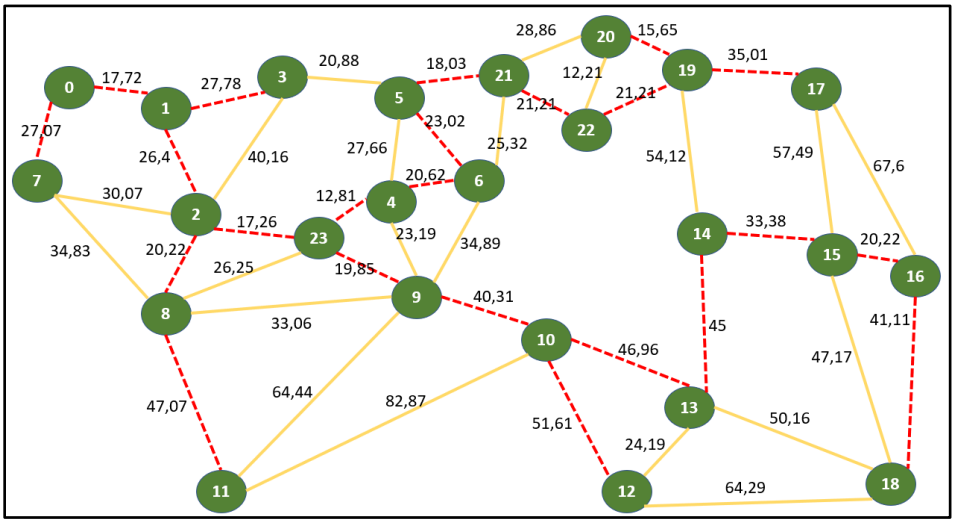
Nota. Adaptado de Weight result. Tomado de Prim´s Algorithm for Optimizing Fiber Optic Trajectory Planning (p.508), por Iqball, Utama, Purba & Purwanto, 2017, International Journal of Scientific Research in Science and Technology.

La tabla 1 muestra los resultados de ponderación basados en sus coordenadas. Los pesos obtenidos serán utilizados por el trabajo consultado.

2.3.1.1.3. Árbol de expansión mínima obtenido.

Figura 2.

Resultado del algoritmo de Prim.



Nota. Prim´s algorithm result. Tomado de Prim´s Algorithm for Optimizing Fiber Optic Trajectory Planning (p.508), por Iqball, Utama, Purba & Purwanto, 2017, International Journal of Scientific Research in Science and Technology.

La figura 2 muestra el resultado obtenido, el cual es el árbol de expansión mínima. Se nota que, si hay ramas, los pesos se seleccionan en base a los pesos más pequeños.

2.3.1.2. Solución resultante en la ejecución del programa implementado en grupo.

Para llevar a cabo la solución del problema señalado anteriormente, con el programa implementado por este equipo, se utilizó los datos proporcionados en la Tabla 1. Cabe señalar que se utilizó la frecuencia relativa de los pesos mas no los pesos como valor entero que nos muestra la Tabla 1, esto se realizó con la finalidad de adecuar el ingreso de datos según el funcionamiento de nuestro programa. Es preciso indicar que el uso de la frecuencia relativa nos ayudara a tener una mejor precisión al obtener el árbol de expansión mínima.

El ingreso de datos se realizó a través de un archivo.txt, este archivo contiene el número de vértices, aristas, los nodos y la frecuencia relativa de los pesos de cada nodo.

Tabla 2.

Tabla de frecuencia relativa de los pesos.

Frecuencia Relativa de los Pesos			
0.00602479	0.00944519	0.00860879	
0.00920379	0.01365438		0.00920379
	0.00709919		0.01022379
0.00602479		0.00940439	0.01184219
0.00897599	0.00701079		
0.00944519	0.00788459		0.00687479
	0.00435539		0.01184219
0.00897599		0.00940439	0.01124039
0.01365438	0.00782679		0.01600378
0.01022379	0.00613019		0.00892499
0.00687479		0.00701079	
0.00586839	0.00782679		0.00788459
	0.01186259		0.01186259
			0.01124039
			0.01370538

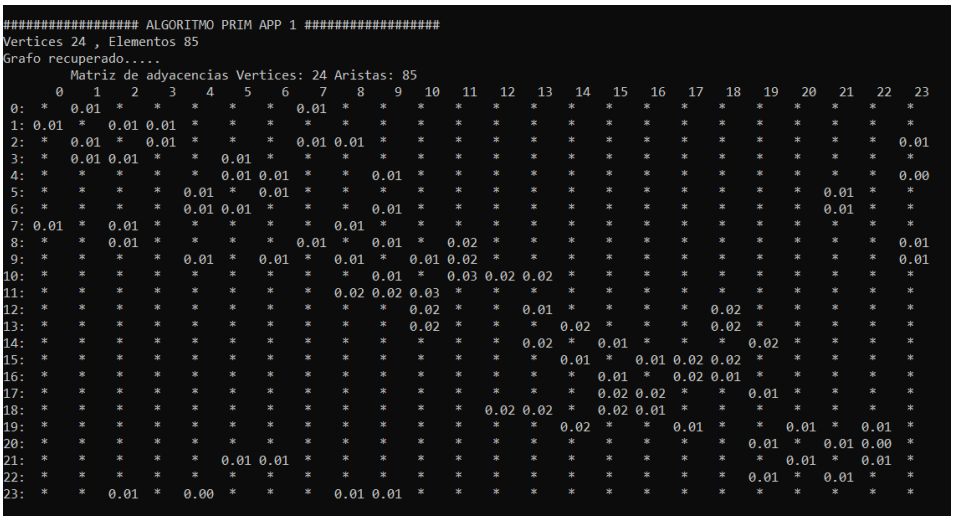
0.02190957	0.01134919	0.01840078
0.00674899	0.01840078	0.01190339
		0.00532099
0.01370538	0.01134919	0.00721139
0.02817577	0.00687479	
0.01754738	0.01954658	0.00532099
0.01596638	0.01603778	0.00981239
		0.00415140
0.01600378	0.00687479	
0.02190957	0.02298397	0.00613019
0.02817577	0.01397738	0.00860879
		0.00981239
0.01754738	0.01954658	0.00721139
0.00822459	0.02298397	
0.02185857	0.01190339	0.00721139
		0.00721139
0.01596638	0.02185857	
0.01529998	0.01705438	0.00586839
0.01705438	0.01603778	0.00435539
	0.01397738	0.00892499
0.01529998		0.00674899

Fuente: Elaboración propia

2.3.1.2.1. Resultados obtenidos.

Imagen 1.

Matriz de adyacencias.



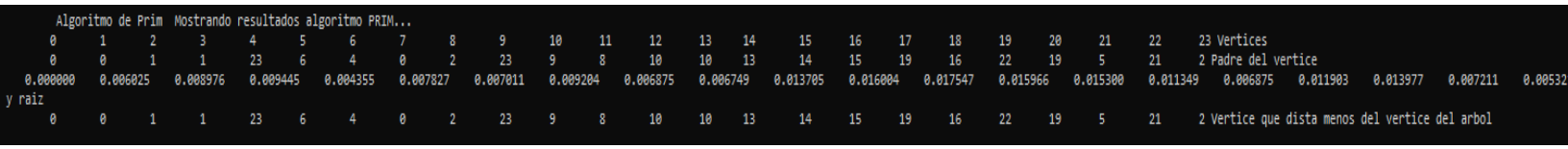
```
################## ALGORITMO PRIM APP 1 #################
Vertices 24 , Elementos 85
Grafo recuperado.....
        Matriz de adyacencias Vertices: 24 Aristas: 85
       0    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20   21   22   23
 0:    *   0.01    *    *    *    *    *   0.01    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *
 1:  0.01   *   0.01 0.01    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *
 2:    *   0.01    *   0.01    *    *    *   0.01 0.01    *    *    *    *    *    *    *    *    *    *    *    *    *   0.01
 3:    *   0.01 0.01    *    *   0.01    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *
 4:    *    *    *    *    *   0.01 0.01    *    *   0.01    *    *    *    *    *    *    *    *    *    *    *    *   0.00
 5:    *    *    *    *   0.01    *   0.01    *    *    *    *    *    *    *    *    *    *    *    *    *   0.01    *    *
 6:    *    *    *    *   0.01 0.01    *    *    *   0.01    *    *    *    *    *    *    *    *    *    *   0.01    *    *
 7:  0.01   *   0.01    *    *    *    *    *   0.01    *    *    *    *    *    *    *    *    *    *    *    *    *    *
 8:    *    *   0.01    *    *    *    *   0.01    *   0.01    *   0.02    *    *    *    *    *    *    *    *    *    *   0.01
 9:    *    *    *    *   0.01    *   0.01    *   0.01    *   0.01 0.02    *    *    *    *    *    *    *    *    *    *   0.01
10:    *    *    *    *    *    *    *    *    *    *   0.01    *   0.03 0.02 0.02    *    *    *    *    *    *    *    *    *
11:    *    *    *    *    *    *    *    *    *    *    *   0.02 0.02 0.03    *    *    *    *    *    *    *    *    *    *
12:    *    *    *    *    *    *    *    *    *    *    *    *   0.02    *    *   0.01    *    *    *    *   0.02    *    *    *
13:    *    *    *    *    *    *    *    *    *    *    *    *   0.02    *    *    *   0.02    *    *    *   0.02    *    *    *
14:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.02    *   0.01    *    *    *   0.02    *    *
15:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.01    *   0.01 0.02 0.02    *    *    *
16:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.01    *   0.02 0.01    *    *    *
17:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.02 0.02    *    *   0.01    *    *
18:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.02 0.02    *   0.02 0.01    *    *    *    *
19:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.02    *    *   0.01    *    *   0.01    *
20:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.01    *   0.01 0.00
21:    *    *    *    *    *   0.01 0.01    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.01    *
22:    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *    *   0.01    *   0.01    *
23:    *    *   0.01    *   0.00    *    *    *   0.01 0.01    *    *    *    *    *    *    *    *    *    *    *    *    *
```

Fuente: Elaboración propia

En la Imagen1 se muestra la Matriz de adyacencias de los datos ingresados a través del archivo.txt. En la Imagen 1 se puede observar que los pesos de los datos, aparentemente, se repiten, pero no es así; puesto que, estos pesos al ser ingresados como frecuencias relativas contienen hasta 8 dígitos en la parte decimal, tal como lo muestra la Tabla 2. El programa implementado solo muestra hasta dos dígitos en la parte decimal del número, es por ello que, el programa, redondea cada frecuencia relativa ingresada, mostrando de esta forma tan solo los decimales permitidos en la matriz de adyacencia. Es preciso especificar que, al mostrar los pesos de esta forma no se alterará los resultados obtenidos, esto se verá en la siguiente página. Consideramos no mostrar todos los decimales para un mejor orden y de esta forma poder visualizar de una manera adecuada la matriz de adyacencia en el programa.
Otro punto importante por señalar es que el grafo contiene 24 vértices y ordenar todos estos datos en una matriz de adyacencia puede resultar tedioso si se colocan todos los valores decimales de sus pesos como en este caso.

Imagen 2.

Aplicación del algoritmo de Prim.



Fuente: Elaboración propia

En la imagen 2 se muestra el resultado al aplicar el algoritmo de Prim con los datos ingresados a través del archivo.txt, cade señalar que los datos ingresados son los proporcionados por el estudio realizado.

En esta imagen se muestra el árbol de expansión mínima obtenido por el programa desarrollado, este resultado muestra los vértices, los padres de cada vértice y los pesos de cada trayecto entre vértice y raíz.

Como se puede observar en la Imagen 2, los pesos tienen 6 cifras decimales, siendo esto un factor por el cual no se pueda mostrar de una forma más ordenada los resultados obtenidos. También, se puede notar que, a diferencia de la matriz de adyacencia, en esta parte de los resultados si se logra observar de forma precisa el valor de cada peso, puesto que es necesario indicar de forma exacta el peso que hay en cada trayecto entre vértice y raíz.

El programa también muestra los vértices que distan en menor longitud del vértice del árbol, cabe señalar que este dato se muestra de forma adicional, puesto que, el estudio no lo considera.

2.3.1.2.2. Comparación del trabajo fuente con la solución grupal.

Observando los resultados obtenidos por el estudio (Figura 2) y los resultados obtenidos por el programa desarrollado (Imagen 2), se puede observar que los resultados obtenidos con el programa desarrollo obtiene como solución un árbol de expansión mínima igual al del estudio. El único cambio que se puede observar es en los pesos, puesto que, como se indicó anteriormente, se utilizó las frecuencias relativas de los pesos que

nos brindan como dato. Sin embargo, se puede observar que los resultados no fueron afectados y esto se puede corroborar ahora que ya fue ejecutado el programa desarrollado y el árbol de expansión mínima obtenido es igual al del estudio.

Finalmente se puede concluir, que el programa implementado se encuentra en buen funcionamiento y puede ser aplicado a estudios similares al mostrado en este informe.

2.3.2. Identificación de áreas aisladas para la prevención y protección de desastres naturales.

En este estudio se exploró las carreteras en el condado de Taitung, China, para encontrar un área aislada para la prevención y protección ante desastres naturales.

Wang, Hsieh, Huang (2018) señalan que este estudio asigna valores de riesgo de comunidades, carreteras e instalaciones clave a los nodos, bordes y pesos de la teoría de grafos y utiliza el algoritmo de Prim en la teoría de grafos para encontrar el árbol de expansión mínimo completo (MST) con valores por debajo del umbral de riesgo predeterminado. Cada MST forma un área separada. Si el área no puede comunicarse con el árbol de expansión primario dentro de un tiempo válido, entonces se determinará que es un área aislada. En otras palabras, el árbol formado por el subconjunto de aristas encontrado por este algoritmo no solo incluye los vértices en el grafico conectado, sino también la suma de los pesos de todas las aristas.

Según lo señalado anteriormente por los autores del estudio, se pudo notar que el algoritmo de Prim forma parte de un conjunto de filtros que permiten llegar al árbol de expansión mínima y por ende identificar las áreas aisladas.

Wang et.al. (2018) concluyen que aún es difícil lograr una alerta integral y un control efectivo de los desastres, a pesar de la innovación y mejora para reducir el alcance de los desastres, debido a la complejidad e incertidumbre de los desastres.

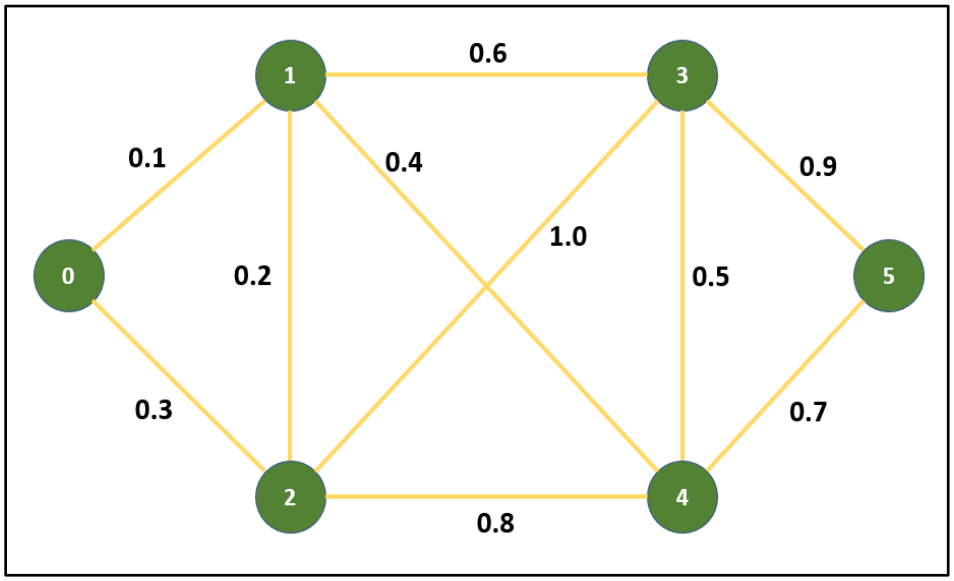
Vertice	Padre del vertice
0	0
1	0
2	1
3	1
4	23
5	6
6	4
7	0
8	2
9	23
10	9
11	8
12	10
13	10
14	13
15	14
16	15
17	19
18	16
19	22
20	19
21	5
22	21
23	2

2.3.2.1. Información proporcionada por el trabajo consultado.

2.3.2.1.1. Grafo del diagrama de red de carreteras del condado de Taitung.

Figura 3.

Grafo de carreteras del condado de Taitung.



Nota. Network diagram. Tomado de Applying Prim´s Algorithm to Identify Isolated Areas for Natural Disaster Prevention and Protection (p. 425), por Wang, Hsieh & Huang, 2018, Scientific Reseach.

En la figura 3 se muestra los vértices que representan los pueblos que se encuentran dentro de la ciudad de Taitung y las aristas representan las carreteras.

2.3.2.1.2. Tabla de pesos según la red de carreteras del condado de Taitung.

Tabla 2.

Tabla de pesos.

Nodo 1	Nodo 2	Peso
[0]	[1]	0.1
[0]	[2]	0.3
[1]	[0]	0.1
[1]	[2]	0.2
[1]	[3]	0.6
[1]	[4]	0.4

[2]	[0]	0.3
[2]	[1]	0.2
[2]	[3]	1
[2]	[4]	0.8
[3]	[1]	0.6
[3]	[2]	1
[3]	[4]	0.5
[3]	[5]	0.9
[4]	[1]	0.4
[4]	[2]	0.8
[4]	[3]	0.5
[4]	[5]	0.7
[5]	[3]	0.9
[5]	[4]	0.7
[5]	[4]	0.7

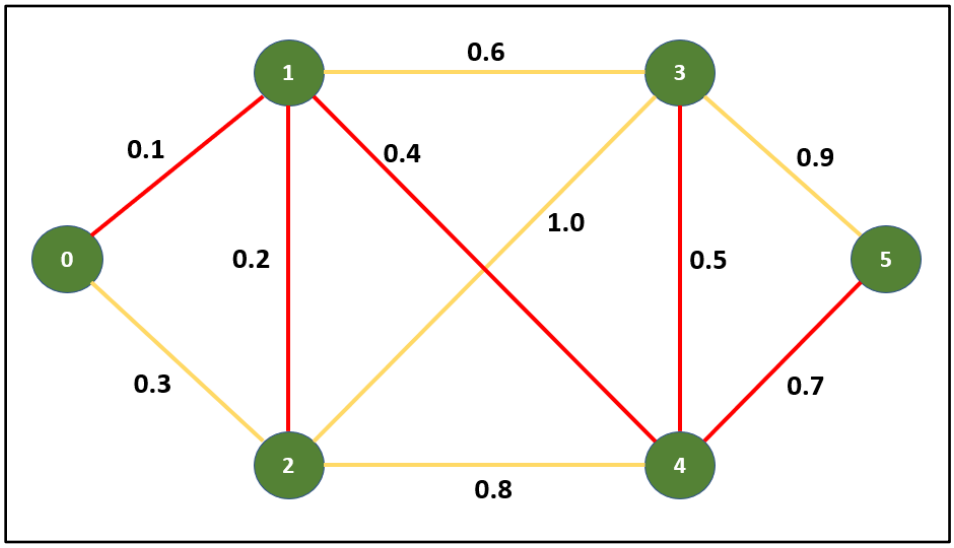
Fuente: Elaboración propia

La tabla 2 muestra los datos proporcionados por el estudio realizado en el condado de Taitung, China.

2.3.2.1.3. Árbol de expansión mínima obtenido.

Figura 4.

Resultado de la aplicación del algoritmo de Prim.



Nota. Network diagram. Tomado de Applying Prim´s Algorithm to Identify Isolated Areas for Natural Disaster Prevention and Protection (p. 425), por Wang, Hsieh & Huang, 2018, Scientific Reseach.

La figura 4 muestra el resultado obtenido del estudio, el cual es un área aislada.

Establecer si el área encontrada es un área aislada o no depende de otros parámetros geográficos que se aplican como filtros para determinar si el árbol de expansión mínimo cumple con los requisitos establecidos.

2.3.2.2. Solución resultante en la ejecución del programa implementado en grupo.

Para llevar a cabo la solución del problema señalado anteriormente, con el programa implementado por este equipo, se utilizó los datos proporcionados en la Tabla 3. Cabe señalar que se utilizó la frecuencia relativa de los pesos mas no los pesos como valor entero que nos muestra la Tabla 3, esto se realizó con la finalidad de adecuar el ingreso de datos según el funcionamiento de nuestro programa. Es preciso indicar que el uso de la frecuencia relativa nos ayudara a tener una mejor precisión al obtener el árbol de expansión mínima.

El ingreso de datos se realizó a través de un archivo.txt, este archivo contiene el número de vértices, aristas, los nodos y la frecuencia relativa de los pesos de cada nodo.

Tabla 4.

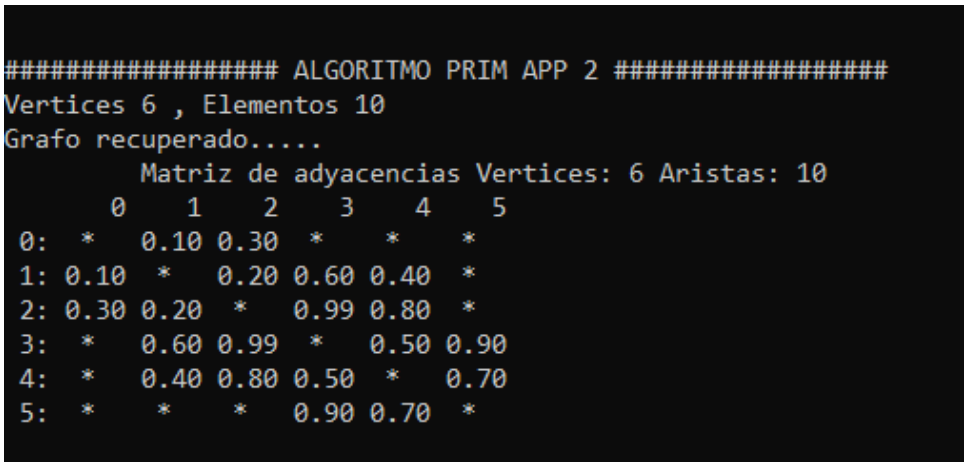
Tabla de frecuencias relativas de los pesos.

Frecuencia Relativa de los Pesos
0.008547
0.025641
0.008547
0.017094
0.051282
0.034188
0.025641
0.017094
0.085470
0.068376
0.051282
0.085470
0.042735
0.076923
0.034188
0.068376
0.042735
0.059829
0.076923
0.059829
0.059829

Fuente: Elaboración propia

2.3.2.2.1. Resultados obtenidos.

Matriz de Adyacencias

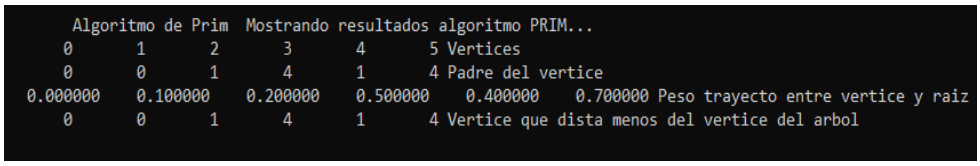


Fuente: Elaboración propia

En la Imagen3 se muestra la Matriz de adyacencias de los datos ingresados a través del archivo.txt. En la Imagen 3 se puede observar que los pesos de los datos, aparentemente, se repiten, pero no es así; puesto que, estos pesos al ser ingresados como frecuencias relativas contienen hasta 6 dígitos en la parte decimal, tal como lo muestra la Tabla 3. El programa implementado solo muestra hasta dos dígitos en la parte decimal del número, es por ello por lo que el programa redondea cada frecuencia relativa ingresada, mostrando de esta forma tan solo los decimales permitidos en la matriz de adyacencia. Es preciso especificar que, al mostrar los pesos de esta forma no se alterará los resultados obtenidos, esto se verá en el siguiente punto. Consideramos no mostrar todos los decimales para un mejor orden y de esta forma poder visualizar de una manera adecuada la matriz de adyacencia en el programa.

Imagen 4

Aplicación del algoritmo de Prim



Fuente: Elaboración propia

En la imagen 4 se muestra el resultado al aplicar el algoritmo de Prim con los datos ingresados a través del archivo.txt, cade señalar que los datos ingresados son los proporcionados por el estudio realizado.

En esta imagen se muestra el árbol de expansión mínima obtenido por el programa desarrollado, el cual representa el área aislada del territorio de Taitung. Este resultado muestra los vértices, los padres de cada vértice y los pesos de cada trayecto entre vértice y raíz.

Se puede notar que, a diferencia de la matriz de adyacencia, en esta parte de los resultados si se logra observar de forma precisa el valor de cada peso, puesto que es necesario indicar de forma exacta el peso que hay en cada trayecto entre vértice y raíz.

El programa también muestra los vértices que distan en menor longitud del vértice del árbol. Según el estudio, estos vértices también pueden ser considerados como una posible área aislada puesto que el árbol de expansión mínimo en una posible área aislada mas no se puede asegurar que lo sea, ya que pasa por una serie de filtros externos al algoritmo.

2.3.2.2.2. Comparación del trabajo fuente con el trabajo grupal.

Observando los resultados obtenidos por el estudio (Figura 4) y los resultados obtenidos por el programa desarrollado (Imagen 4), se puede observar que el resultado obtenido con el programa desarrollado muestra como solución un árbol de expansión mínima, el cual representa una posible área aislada, igual al del estudio. El único cambio que se puede observar es en los pesos, puesto que, como se indicó anteriormente, se utilizó las frecuencias relativas de los pesos que nos brindan como dato.

Sin embargo, se puede observar que los resultados no fueron afectados y esto se puede corroborar ahora que ya fue ejecutado el programa desarrollado y el árbol de expansión mínima, es decir, el resultado del recorrido de la posible área aislada obtenido es igual al del estudio.

Finalmente se puede concluir, que el programa implementado se encuentra en buen funcionamiento y puede ser aplicado a estudios similares al mostrado en este informe. Si bien es cierto el algoritmo solo hace una parte del trabajo al identificar una posible área aislada; sin embargo, no se puede negar que cumple un rol importante al encontrar dicha área para su evaluación y también posibles vértices a tomar en cuenta para su estudio. Cabe resaltar que en el programa solo se colocan datos previamente obtenidos de un estudio realizado como, por ejemplo, un estudio de coordenadas de un territorio determinado para obtener las distancias de varios puntos.

2.4. Algoritmo de Kruskal.

2.4.1. Búsqueda del camino más corto a la ubicación de una tienda de construcción en la ciudad de Bogor, Indonesia.

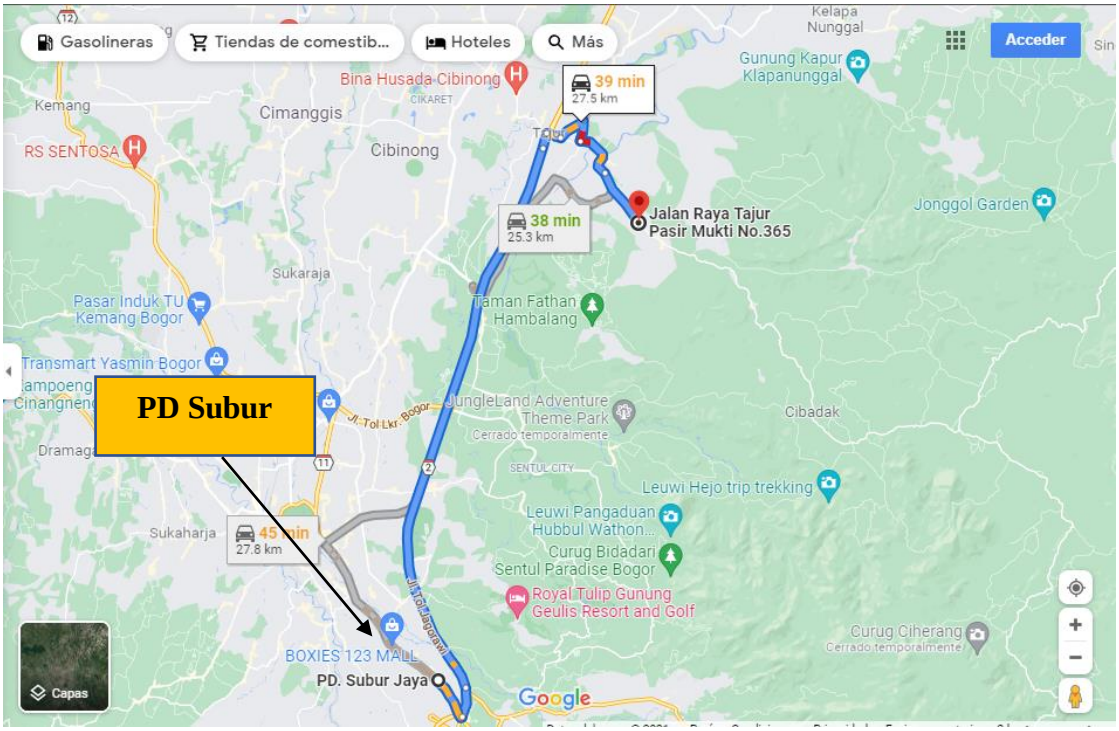
En el siguiente caso de estudio dado por Erniyati (2019), se menciona que encontrar el camino más corto entorno al envío de mercancías por tierra es una cuestión importante en el sector empresarial; sin embargo, en un país tan caótico en cuestión a carreteras, tráfico y trayectorias como Indonesia, específicamente, la ciudad de Bogor, la determinación de la trayectoria más eficiente puede minimizar los costos de envío y tiempo, por lo cual, es necesario determinar un algoritmo o sistema que busque la ruta más corta entre las varias existentes y posibles según el tráfico. De este modo, se incluye al algoritmo de Kruskal, ya que, es uno de los más utilizados en la resolución de problemas de árboles de expansión mínimo, en los cuales, se elige el enlace que tiene el menor peso de cada nodo con la condición de que los enlaces seleccionados no puedan formar una estructura

circular. Así, para el presente caso de estudio, el autor propone utilizar los datos de actividades controlados por transporte terrestre a fin de que, por la implementación del algoritmo de Kruskal, se encuentre la ruta más cercana al almacén del edificio para el posterior envío de mercancías, tal que, la conclusión del trabajo define que la ruta generada por el algoritmo fue 83% más reducida que la que propone Google Maps.

Así, a fin de determinar el camino más cercano a la ubicación de una tienda en la ciudad de Bogor, Indonesia desde ‘s’, sea el punto de partida, desde Kranggan No. 100 Kerangaan Road, Gunung Putri, Bogor hacia 12 tiendas de PD. Subur, que es la tienda de materiales para la construcción en Surabaya, Indonesia, con el siguiente mapa obtenido:

Figura 1.

Mapa que define la ubicación de PD. Subur con la ruta Jl. Tolkr. Bogor



Se tiene a la ruta Jl. Tolkr, Bogor, que une a la tienda de materiales en mención con las diversas 12 locaciones propuestas, las cuales, se destinarán a continuación:

Tabla 1.

12 tiendas destino del centro de materiales con sus direcciones exactas.

p1 (TB Subur Jaya, Jl. Raya Tajur No. 315 Kec. Bogor Timur, Kota)	**p7** (TB. Pandu Makmur , Jl. Tegal Gundil Kec. Bogor Utara)
p2 (TB. Hutan Mas, Jl. Raya Tajur Ciawi Kec. Bogor Timur)	**p8** (TB. Jawa Jl. Raya Bogor Kedung Halang Km. 55 No. 225 Kec Bogor Utara)
p3 (TB. Kurnia Jl. Raya Tajur No. 63A Pakuan, Kec. Bogor Selatan	**p9** (TB. Abadi Jaya Toko, Jl. Sindang Barang No. 61 Gunung Batu, Kec. Bogor Barat)
p4 (TB. Setia Makmur, Jl. Pahlawan No. 183 Bondongan, Kec, Bogor Selatan)	**p10** (TB. Sarana Bangunan, Jl. Raya Semplak No. 196 Atang Senjaya Kemang, Kec. Bogor Barat)
p5 (TB. Prapatan Tegal Lega, Jl. Tegal Lega No. 1, Kec. Bogor Tengah	**p11** (TB. Wahana Agung , Jl. Raya Cibadak - Ciampea Cibadak Tanah Sareal K
p6 (TB. Setia Abadi, Jl. Pengadilan No. 1 DD Kec. Bogor Tengah)	**p12** (TB. Purba , Jl. Kayu Manis No. 85 Cibadak Kec. Tanah Sareal Kota Bogor

Se dan los 12 almacenes destino correspondientes a la tienda PD. Subur, de modo que, con sus direcciones, el autor propone distancias estimadas dadas en el siguiente apartado.

2.4.1.1. Información proporcionada por el trabajo consultado.

2.4.1.1.1. Grafo que incluye distancias ordenadas.

El autor propone que ‘s’, que como se mencionó anteriormente, es el punto inicial, y, los 12 almacenes sean dados con abreviaturas con p añadido del número de almacén; sin embargo, de acuerdo a lo aceptado por el programa dado en el presente informe, estas abreviaturas serán distinguidas convenientemente por números enteros del 0 al 11, tal que, el grafo contenga 11 nodos y 14 aristas. No obstante, el autor propone una ruta aún más reducida y que va desde ‘s’ hasta p8, no hasta p12.

Tabla 2.

Tabla original del autor que incluye las aristas con distancias.

N °.	Nodo 1	Nodo 2	Distancias (km)
1	s	a	500
2	s	c	7000
3	a	b	3300
4	a	d	4800
5	b	c	450
6	b	d	2400
7	c	f	5800
8	d	e	5500
9	e	g	8000
10	f	g	2500
11	f	h	3200
12	g	P8	4800
13	h	i	4600
14	i	P8	1200

Tabla 3.

Tabla modificada con las aristas y sus distancias (elaboración propia)

N °.	Nodo 1	Nodo 2	Distancias (km)
1	10	0	500
2	10	2	7000
3	0	1	3300
4	0	3	4800
5	1	2	450
6	1	3	2400
7	2	5	5800
8	3	4	5500
9	4	6	8000
10	5	6	2500
11	5	7	3200

12	6	9	4800
13	7	8	4600
14	8	9	1200

Así, se proponen las equivalencias entre la tienda matriz PD. Subur y los 12 almacenes de la misma, tal que, se destinen las abreviaturas siguientes:

- **s:** 10 (PD. Subur)
- **a:** 0
- **b:** 1
- **c:** 2
- **d:** 3
- **e:** 4
- **f:** 5
- **g:** 6
- **h:** 7
- **i:** 8
- **P8:** 9

2.4.1.1.2. Solución definida por el autor.

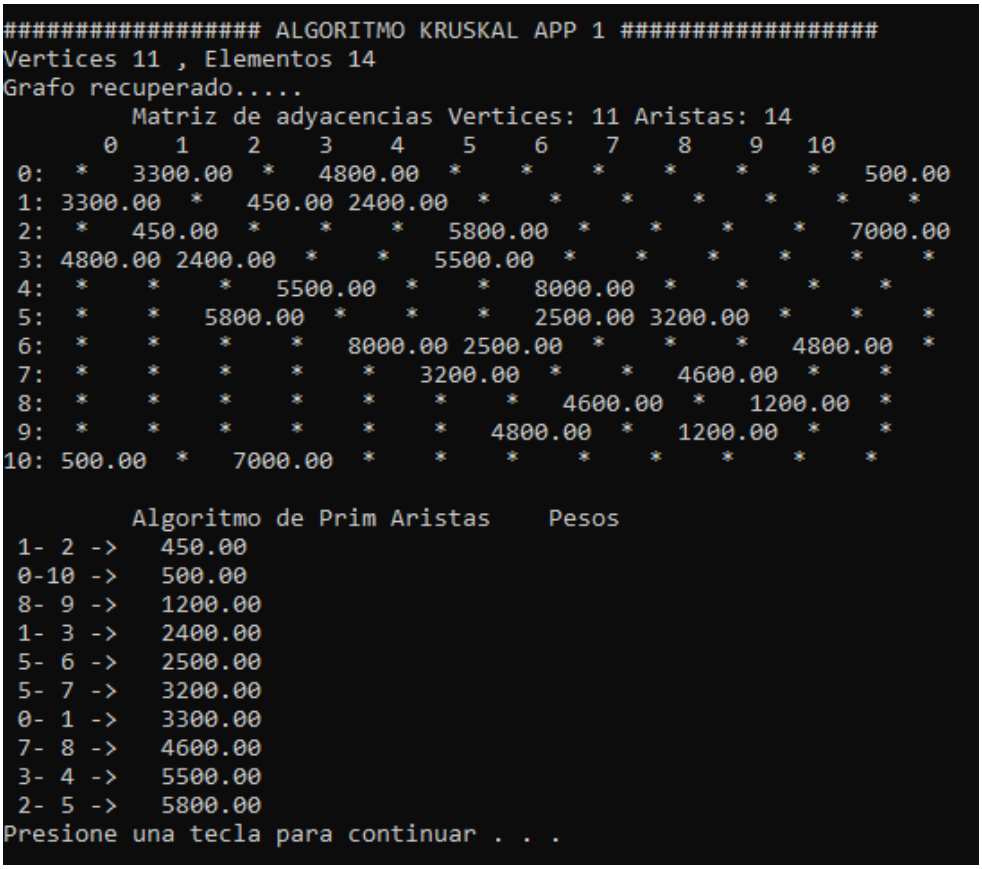
El autor propone una solución con 14 pasos, sean los siguientes:

- **Paso 1:** b-c o 1-2.
- **Paso 2:** s-a o 10-0.
- **Paso 3:** i-P8 o 8-9.
- **Paso 4:** b-d o 1-3.
- **Paso 5:** f-g o 5-6.
- **Paso 6:** f-h o 5-7.
- **Paso 7:** a-b o 0-1.
- **Paso 8:** h-i o 7-8.
- **Paso 9:** a-d o 0-3 forman un círculo, por lo que solo se señala.
- **Paso 10:** g-p8 forman un círculo, por lo que solo se señala.
- **Paso 11:** d-e o 3-4.

- **Paso 12:** c-f o 2-5.
- **Paso 13:** a-c o 0-2.
- **Paso 14:** e-g o 4-6 forman un círculo, por lo que solo se señala.

Finalmente, el autor no proporciona con claridad un valor de kilómetros para la ruta mínima; no obstante, considera mencionar que los resultados de las pruebas del algoritmo de Kruskal en contraste con los propuestos por Google Maps a partir de las direcciones de los 12 almacenes, da cuenta que la comparación de la distancia de la ruta encontrada por el algoritmo de Kruskal es mejor en 83% que la encontrada por Google, por lo que, menciona al lector en general que, espera que se desarrolle una futura investigación agregando el tiempo de viaje y las condiciones pero en tiempo real y así, comparar iterativamente otros métodos de rutas más cortas incluso, dependiendo de las situaciones víales por el tráfico dado.

2.4.1.2. Solución resultante en la ejecución del programa implementado en grupo.



```
################## ALGORITMO KRUSKAL APP 1 ##################
Vertices 11 , Elementos 14
Grafo recuperado.....
        Matriz de adyacencias Vertices: 11 Aristas: 14
      0    1    2    3    4    5    6    7    8    9   10
 0:  *   3300.00  *   4800.00  *    *    *    *    *    *   500.00
 1: 3300.00  *   450.00 2400.00  *    *    *    *    *    *    *
 2:  *   450.00  *    *    *   5800.00  *    *    *    *   7000.00
 3: 4800.00 2400.00  *    *   5500.00  *    *    *    *    *    *
 4:  *    *    *   5500.00  *    *   8000.00  *    *    *    *
 5:  *    *   5800.00  *    *    *   2500.00 3200.00  *    *    *
 6:  *    *    *    *   8000.00 2500.00  *    *    *   4800.00  *
 7:  *    *    *    *    *   3200.00  *    *   4600.00  *    *
 8:  *    *    *    *    *    *    *   4600.00  *   1200.00  *
 9:  *    *    *    *    *    *   4800.00  *   1200.00  *    *
10: 500.00  *   7000.00  *    *    *    *    *    *    *    *

        Algoritmo de Prim Aristas    Pesos
 1- 2 ->   450.00
 0-10 ->   500.00
 8- 9 ->   1200.00
 1- 3 ->   2400.00
 5- 6 ->   2500.00
 5- 7 ->   3200.00
 0- 1 ->   3300.00
 7- 8 ->   4600.00
 3- 4 ->   5500.00
 2- 5 ->   5800.00
Presione una tecla para continuar . . .
```

Mediante la ejecución del programa planteado, se adjunta la solución por el algoritmo de Kruskal del presente caso en donde se incluye la cantidad de vértices que son 11; las aristas, que son 14, la matriz de adyacencia y los 10 enlaces dispuestos como conexiones de las rutas entre los almacenes de la tienda PD. Subur en Indonesia y el destino de la misma, donde, al realizar el siguiente cálculo: 450+500+1200+2400+2500+3200+3300+4600+5500+5800=29450 km, tal que, en definitiva, esta sea la ruta más corta que define la conexión entre los almacenes de la tienda y justamente la tienda y así, de lo dado como inicial por el autor, se conseguirán reducir seguramente los tiempos y costes en las entregas de los materiales, condición que es imprescindible en el abastecimiento de una tienda de tal relevancia en la ciudad de Bogor.

2.4.2. Optimización de costos totales para la instalación de una red de cable de fibra óptica que considere la conexión a diversas ciudades de cierto país.

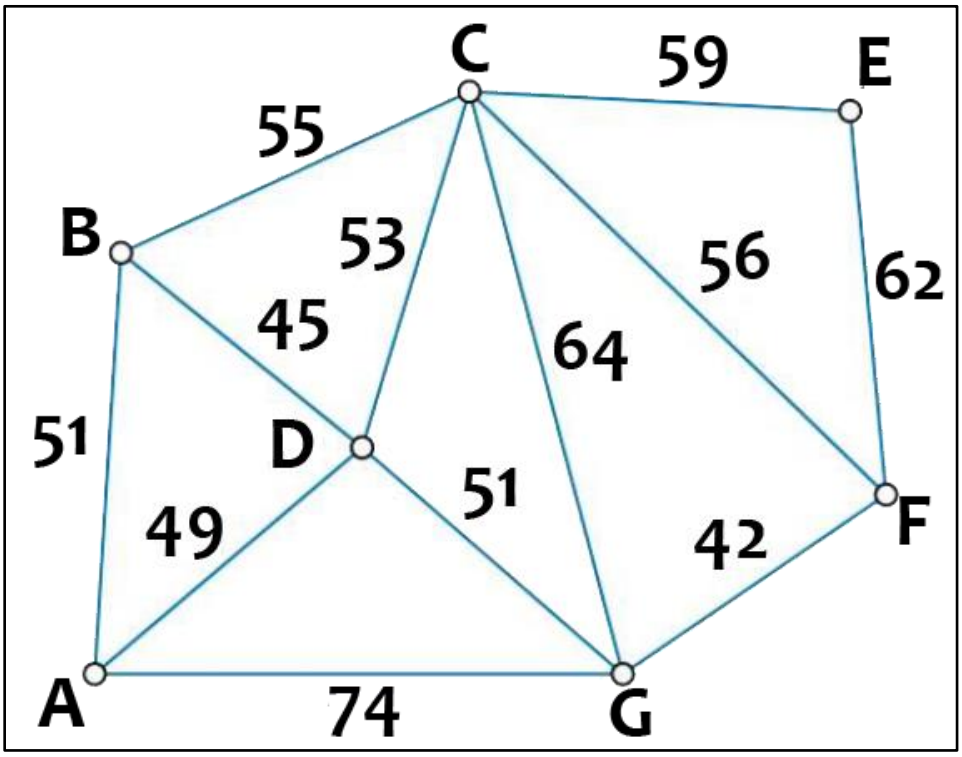
En el siguiente caso de estudio dado por Tannenbaum (1991), se propone la búsqueda e implementación de un método que destine la optimización de costos totales en la instalación de una red de cable de fibra óptica con la finalidad de considerar la conexión entre ciudades de un cierto país, sea que, particularmente, se definan 7, tales como Agua Linda, Boas, Caimán, Delicias, El Dorado, Frutal y Guaraní, las cuales, por condición del caso, presentan 1 sola torre por cada ciudad, de modo que, se pretenda minimizar el costo total que implica el proyecto que permita la conexión dada en redes de cableado óptimo.

Así, se considera la solución del presente caso mediante la búsqueda del árbol mínimo de expansión de la red a diseñar, esto, aplicando el algoritmo de Kruskal, donde, se estime que cada 1 de las 7 torres presentes sean dadas como vértices o nodos y, los pesos asociados a cada uno sean dados como el costo en millones de dólares, los cuales, resultan de las uniones entre cada uno por medio de conexiones de cables dados como aristas, teniendo en cuenta que, por requerimiento de la solución en consideración de la especificación del programa implementado, los nodos originales serán tratados como números enteros y no como las abreviaturas en lexemas de un solo caracter, tal como se propone en el análisis y solución dados en el libro, sea que, de esta manera, se detalle lo mencionado en el libro para que, luego, se exponga la solución resultante de la ejecución del programa y, finalmente, se comparen las 2 a fin de establecer conclusiones necesarias.

2.4.2.1. Información proporcionada por el trabajo consultado.

2.4.2.1.1. Grafo de las conexiones posibles entre ciudades y el costo.

Figura 1.*Rutas y costos de las conexiones entre ciudades del proyecto.*



Se denota al grafo original del texto propuesto con 7 nodos y 12 aristas, tal que, además, se refieran las equivalencias de las ciudades abreviadas a lexemas de un solo caracter por fines prácticos, tal como sigue:

- Agua Linda: A
- Boas: B
- Caimán: C
- Delicias: D
- El Dorado: E
- Frutal: F
- Guaraní: G

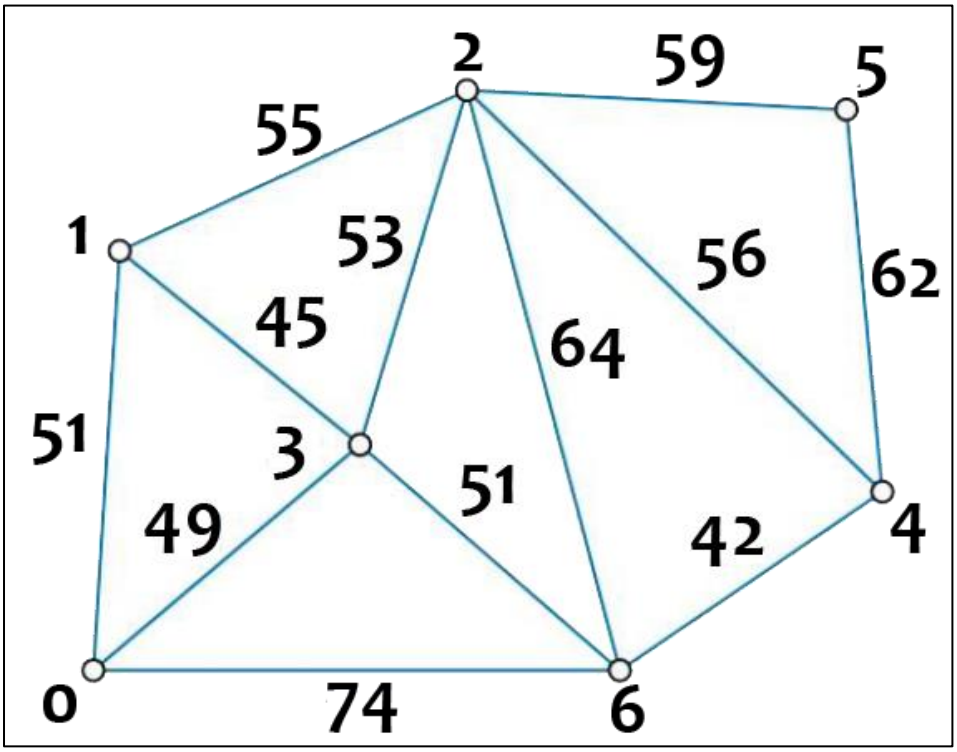
Además, se definen a las aristas consideradas como los cables que unen las torres mencionadas y sus respectivos pesos dados como el costo de dichas uniones de cables entre las torres, en millones de dólares.

No obstante, en consideración del programa propuesto en el siguiente proyecto, se debe modificar las abreviaturas dadas en números enteros que consideren desde el 0 en adelante; en específico, del 0 al 6.

Figura 2.

Rutas y costos de las conexiones entre ciudades del proyecto

(Grafo modificado en abreviaturas de las 7 ciudades presentadas)



De esta manera, el grafo modificado será el presentado y así, podrá ser introducido y analizado mediante el algoritmo de Kruskal en el programa, tal que, se consideren las siguientes asociaciones finales como siguen:

- Agua Linda: A (0)
- Boas: B (1)
- Caimán: C (2)
- Delicias: D (3)
- El Dorado: E (4)
- Frutal: F (5)
- Guaraní: G (6)

2.4.2.1.2. Tabla de pesos según la red de cableado entre ciudades.

Tabla 1.

Tabla de pesos de cada arista dada como torre por ciudades.

Nodo 1	Nodo 2	Peso
0	1	51
0	3	49
0	6	74
1	2	55
1	3	45
2	3	53
2	4	59
2	5	56
2	6	64
3	6	51
4	5	62
5	6	42

Se considera cada costo de unión por cableado entre las torres de las 7 ciudades, tal que, se definan, en el modelo de teoría de grafos, las aristas, teniendo en cuenta los 2 nodos de comunicación y sus pesos respectivos.

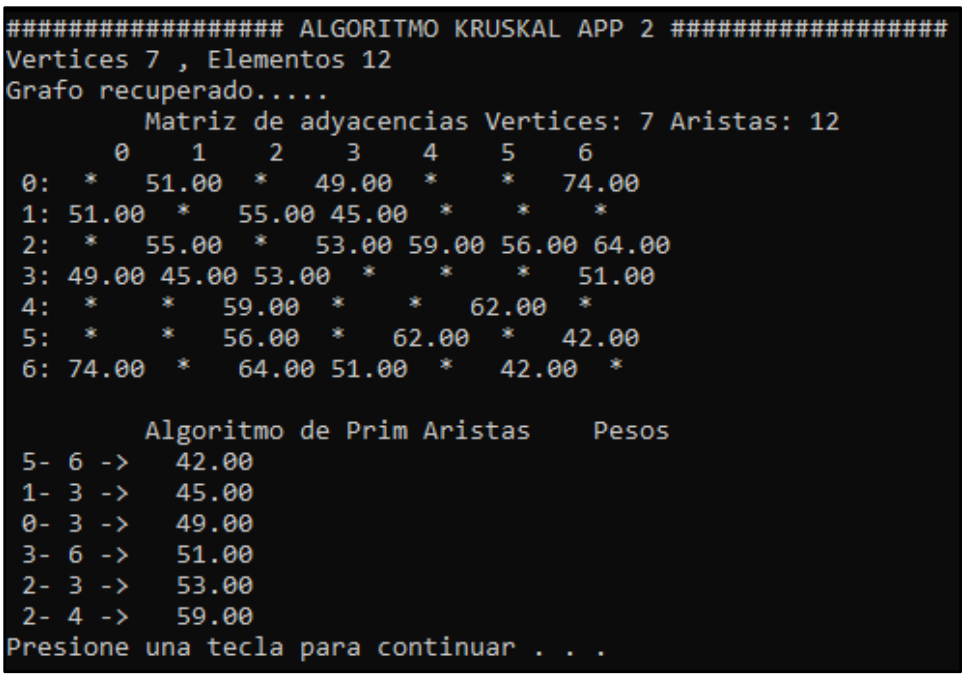
2.4.2.1.3. Solución por pasos definidos por el autor.

En la solución definida por el autor, se consideran 7 pasos a fin de determinar el árbol de expansión mínimo del grafo, o lo que es equivalente, la ruta que propone el menor costo posible a fin de conectar las 7 torres de las ciudades dadas por cableado de red de fibra óptica.

- **Paso 1:** Entre todos los nodos posibles, se deberá elegir el de menor costo, sea justamente el que equivale a 42 millones de dólares, tal que, este nodo será el enlace del MST (árbol mínimo de expansión)

- **Paso 2:** El siguiente enlace más barato disponible es B-D o 1-3 con 45 millones de dólares, con lo cual, se elige para el MST.
- **Paso 3:** El siguiente enlace más barato disponible es A-D 0-3 con 49 millones de dólares y, nuevamente, se elige para el MST.
- **Paso 4:** Para el siguiente enlace más barato, existe un empate entre A-B o 0-1 y D-G o 3-6, ambos en 51 millones de dólares, pero se puede descartar 0-1 y habilitar el otro enlace 3-6 como parte del MST.
- **Paso 5:** El siguiente enlace más barato disponible es C-D o 2-3 a 53 millones de dólares, por lo cual, se agrega como parte del MST.
- **Paso 6:** El siguiente enlace más barato disponible es B-C o 1-2 a 55 millones de dólares, sin embargo, este enlace generaría un nuevo circuito, así que es descartado. Ahora, se verifica la opción de C-F o 2-5 en 56 millones de dólares; sin embargo, este enlace también generaría un nuevo circuito, por lo cual, también es descartado.
  De este modo, la única opción viable es C-E o 2-4 a 59 millones de dólares y se elige, a fin de ser parte del MST propuesto.
- **Paso 7:** Ahora bien, el MST buscado ha sido generado debido a que ya existen 6 enlaces necesarios en un grafo de 7 vértices, por lo cual, la suma total de todos los costes generados es de 299 millones de dólares, siendo esta, la menor posible para la ejecución del proyecto.

2.4.2.2. Solución resultante en la ejecución del programa implementado en grupo.



Mediante la ejecución del programa, se adjunta la solución por el algoritmo de Kruskal del presente caso planteado en donde se incluye la cantidad de vértices que son 7; las aristas, que son 12, la matriz de adyacencia y los 6 enlaces dispuestos como conexiones por cable de las redes de fibra óptica para las ciudades de cierto país en mención, donde, al realizar el siguiente cálculo: 42+45+49+51+53+59=299, resulta la misma cantidad que fue determinada en la solución propuesta por el autor, por lo cual, se considera que ambas guardan relación, que el programa propuesto en el presente informe funciona adecuadamente y que, finalmente, cada enlace descrito coincide; ello, por mención a que, no solo se debe tener en cuenta el resultado de una operación, sino, el fundamento de los pasos por los cuales se obtiene una respuesta, entendiéndose que, pueden existir procedimientos totalmente distintos y ciertamente, alguno de ellos erróneo, y finalmente, llegar a la misma solución, lo cual, claramente no ocurre tal que cada enlace coincide respectivamente como se visualiza.

3. **Conclusiones.**
4. **Referencias bibliográficas.**

Erniyati, P. (2019). *The implementation of the Kruskal algorithm for the search for the shortest path to the location of a building store in the city of Bogor* [Implementación del algoritmo de Kruskal para la búsqueda del camino más corto hacia la ubicación de una tienda de construcción en la ciudad de Bogor]. *IOP Conference Series: Materials Science and Engineering, 621* (012010), 30-31. doi: 10.1088/1757-899X/621/1/012010

Fernández, J.(2018). Cálculo de trayectos mediante algoritmos de búsqueda informada sobre grados ponderados no dirigidos. Universidad Politécnica de Madrid.

Meneses, E. & Arias, E. (2017). Algoritmos alternos de bajo coste para la comparación de rutas metabólicas en plantas.

Mirzaeinia, A., Shahmoradi, J., Roghanchi, P., & Hassanalian, M. (2019). Autonomous routing and power management of drones in gps-denied environments through dijkstra algorithm. In *AIAA Propulsion and Energy 2019 Forum* (p. 4462).

Tannenbaum, P. (1991). *Excursions in Modern Mathematics* [Excursiones en matemáticas modernas]. California, Estados Unidos de América: Pearson Education.

Wang, W.-C., Hsieh, M.-C. y Huang, C.-H. (2018) Aplicación del algoritmo de Prim para identificar áreas aisladas para la prevención y protección de desastres naturales. Ingeniería, 10, 417-431. https://doi.org/10.4236/eng.2018.107029

Xu, Y., Wang, Z., Zheng, Q., & Han, Z. (2012, August). The application of Dijkstra's algorithm in the intelligent fire evacuation system. In *2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics* (Vol. 1, pp. 3-6). IEEE.