DESARROLLO DE ALGORITMO PARA AUMENTAR LA SEGURIDAD EN LAS CALLES DE MEDELLÍN

Jacobo Zuluaga Jaramillo
Universidad Eafit
Colombia
jzuluagaj@eafit.edu.co

Facundo Villa
Universidad Eafit
Uruguay
fvillas@@eafit.edu.co

Santiago Henao
Universidad Eafit
Colombia
shenaor4@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

Andrea Serna
Universidad Eafit
Colombia
asernac1@eafit.edu.co

RESUMEN

Para nadie es un secreto que caminar por algunas calles de Medellín es una actividad bastante riesgosa para el ciudadano, y es que las cifras de acoso y otros delitos se han disparado durante los últimos tiempos. Es importante buscar un método para disminuir las cifras de estos delitos que se cometen en las calles de Medellín, para que de esta manera los ciudadanos se sientan tranquilos en tomar un camino en el que se pueda prevenir el crimen. En este artículo daremos solución a la problemática mencionada a través del algoritmo dijkstra para averiguar tres caminos, donde se dará como resultado uno más corto, otro más eficiente y otro que promedia estas variables para llegar a un destino, probamos el algoritmo poniendo el punto de origen en la Universidad EAFIT y el de destino en la Universidad Nacional, dándonos los siguientes resultados: camino 1; distancia : 9700.925m, riesgo: 0.375, tiempo: 0.103s, camino 2; distancia: 7474.98m, riesgo: 0.342, tiempo: 0.070s, camino 3; distancia: 10128.88m, riesgo: 0.153, tiempo: 0.047s, concluyendo que el proyecto da solución a la problemática arrojando 3 caminos útiles para ir desde un punto a otro, en un tiempo razonable.

Palabras clave

Camino más corto, acoso sexual callejero, identificación de rutas seguras, prevención del crimen.

1. INTRODUCCIÓN

En la actualidad, unas de las problemáticas más visibles en la sociedad es el acoso sexual, este se presnta en todos los ambitos y en todos los espacios, pero en este proyecto buscaremos darle una foco a el acoso sexual en espacios publicos, esta problematica afecta una gran cantidad de las personas, en el caso de las mujeres, se afirma que, 9 de cada 10 mujeres son victimas de acoso sexual callejero y aunque en los hombres la cantidad de acoso disminuya, no quita que se pueda hacer presente. La gran cantidad de afectados por esta problemática es uno de los motivos por el cual decidimos desarrollar este proyecto, que busca darle una solución o al menos una alternativa a las personas que desean transitar por las calles de Medellín sin correr el riesgo de ser afectadas por el acoso sexual callejero.

1.1. Problema

El problema a resolver trata en hallar tres caminos diferentes, que van desde un punto de origen hasta un punto de llegada, estos con el fin de encontrar el camino que mas reduzca las distancias y el riesgo de acoso sexual, este problema causa un impacto en la sociedad y aunque este aplica en toda la poblacion, la parte mas bulnerable es en especial el genero femenino, debido a la gran cantidad de acoso que estas sufren al trancitar por las calles, y seria util darle una solucion para que la persona pueda transitar por un camino seguro sin que este tenga que recorer una gran distancia.

1.2 Solución

Para la solución al problema planteado utilizamos el algoritmo Dijkstra. Este método nos ayudará no solo a encontrar los caminos, sino que también nos ayudará a visualizar cual de estos es el más accesible para el tránsito de los ciudadanos, además que nos dirá cual es el más corto y cual es el más seguro. El algoritmo Dijsktra es, a su vez, el algoritmo más eficiente para el tipo de problema que nos encontramos, ya que este tiene una complejidad de n-1 llamados ($O(n^2)$.

1.3 Estructura del artículo

A continuación, en la Sección 2, presentamos trabajos relacionados con el problema. Posteriormente, en la Sección 3, presentamos los conjuntos de datos y los métodos utilizados en esta investigación. En la Sección 4, presentamos el diseño del algoritmo. Después, en la Sección 5, presentamos los resultados. Finalmente, en la Sección 6, discutimos los resultados y proponemos algunas direcciones de trabajo futuro.

2. TRABAJOS RELACIONADOS

A continuación, explicamos cuatro trabajos relacionados con la búsqueda de caminos para prevenir el acoso sexual callejero y la delincuencia en general.

Explique cuatro (4) artículos relacionados con el problema descrito en el apartado 1.1. Puede encontrar los problemas relacionados en revistas científicas. Considere Google Scholar para su búsqueda. *(En este semestre, el trabajo relacionado es la búsqueda de caminos para prevenir el acoso sexual callejero y la delincuencia en general).*

2.1 Desarrollo de un sistema para la optimización de rutas de trabajo utilizando el algoritmo de Dijkstra y diagramas de Voronoi

Este trabajo de investigación tiene como problemática la obtención de rutas óptimas de trabajo de los diferentes caminos recorribles que existen y que hacen posible el acceso de un lugar a otro. como resultado, usando el algoritmo de Dijkstra, logran la solución de este problema a través de un sistema de optimización [1]

2.2 Algoritmos para calcular la ruta más corta en la malla vial de la ciudad de Bogotá

En este proyecto se propone un sistema de algoritmos, usando A-estrella y Dijkstra, que permita encontrar la ruta más corta entre dos puntos dentro de Bogotá. está utilizando la malla vial existente en la ciudad de Bogotá.[2]

2.3 Análisis comparativo de algoritmos buscadores de caminos A*, Dijkstra y BFS en un juego de Maze Runner

En este proyecto se hace una comparación entre 3 de los algoritmos más conocidos para búsqueda de caminos. En este se pone a prueba la velocidad de cada uno de ellos para ir pasando por varias casillas para que, al final, llegue a la meta con el mayor número de puntos posibles. Para este tipo de juego específico el mejor algoritmo basado en el tiempo y tamaño fue el A*.[3]

2.4 Determinación del Método Óptimo de Operaciones de Ensamble Bimanual con el Algoritmo de Dijkstra (o de Caminos Mínimos)

En este artículo se quiere mejorar la productividad laboral a través del análisis de operaciones. Para probarlo se diseña un caso de estudio para simular un ensamble bimanual, en donde todas las formas de ensamble fueron reemplazadas por nodos obtenidos, entre otras cosas, por la distancia entre ellas. Se utilizó el algoritmo Dijkstra, donde se vió un aumento de la productividad hasta del 20% con el método óptimo.[4]

3. MATERIALES Y MÉTODOS

En esta sección, explicamos cómo se recogieron y procesaron los datos y, después, diferentes alternativas de algoritmos de caminos que reducen tanto la distancia como el riesgo de acoso sexual callejero.

3.1 Recogida y tratamiento de datos

El mapa de Medellín se obtuvo de *Open Street Maps* (OSM)[1] y se descargó utilizando la API[2] OSMnx de Python. El mapa incluye (1) la longitud de cada segmento, en metros; (2) la indicación de si el segmento es de un solo sentido o no, y (3) las representaciones binarias conocidas de las geometrías obtenidas de los metadatos proporcionados por OSM.

Para este proyecto, se calculó una combinación lineal (CL) que captura la máxima varianza entre (i) la fracción de hogares que se sienten inseguros y (ii) la fracción de hogares con ingresos inferiores a un salario mínimo. Estos datos se obtuvieron de la encuesta de calidad de vida de Medellín, de 2017. La CL se normaliza, utilizando el máximo y el mínimo, para obtener valores entre 0 y 1. La CL se obtuvo mediante el análisis de componentes principales. El riesgo de acoso se define como uno menos la CL normalizada. La Figura 1 presenta el riesgo de acoso calculado. El mapa está disponible en GitHub[3].

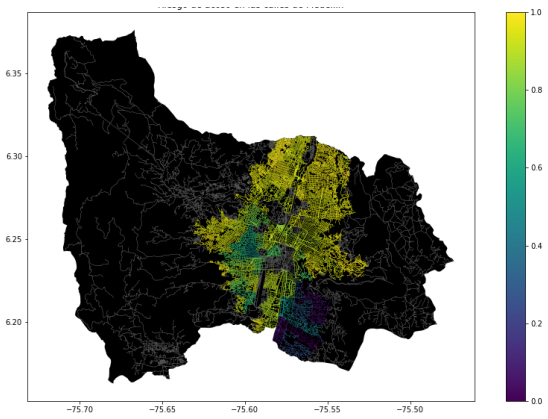


Figura 1. Riesgo de acoso sexual calculado como una combinación lineal de la fracción de hogares que se sienten inseguros y la fracción de hogares con ingresos inferiores a un salario mínimo, obtenidas de la Encuesta de Calidad de Vida de Medellín, de 2017.

[1] https://www.openstreetmap.org/

[2] https://osmnx.readthedocs.io/

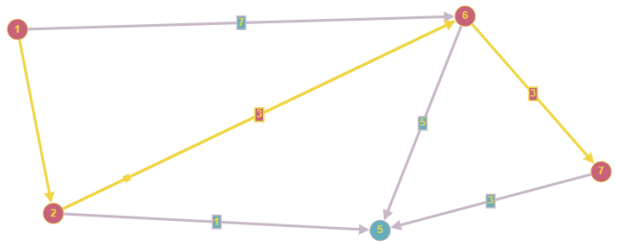
[3]https://github.com/mauriciotoro/ST0245Eafit/tree/master/proyecto/Datasets/

3.2 Alternativas de caminos que reducen el riesgo de acoso sexual callejero y distancia

A continuación, presentamos diferentes algoritmos utilizados para un camino que reduce tanto el acoso sexual callejero como la distancia. *(En este semestre, ejemplos de dichos algoritmos son DFS, BFS, Dijkstra, A*, Bellman, Floyd, entre otros).*

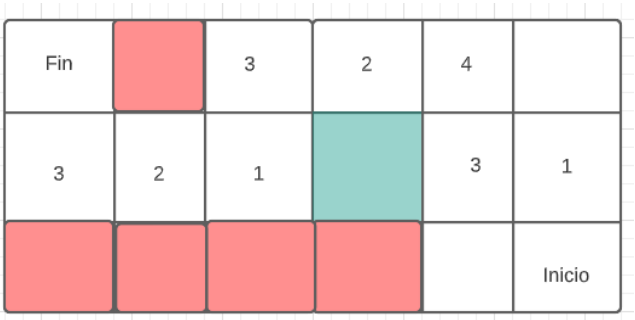
3.2.1 Algoritmo Dijkstra

El algoritmo Dijkstra también conocido como algoritmo de caminos mínimos, se usa para encontrar el camino más corto desde un vértice inicial (origen), hasta los diferentes vértices que se pueden llegar a encontrar o una meta definida, cada arista tiene un peso (como se muestra en la figura) El algoritmo una vez haya encontrado el camino más corto desde el vértice inicio hasta el objetivo, este se detiene. Este algoritmo lo podemos aplicar para la solución de nuestro problema, cada arista sería la distancia multiplicada por el riesgo y el camino con menos peso será el más óptimo, este algoritmo tiene una complejidad de n^2.[5]



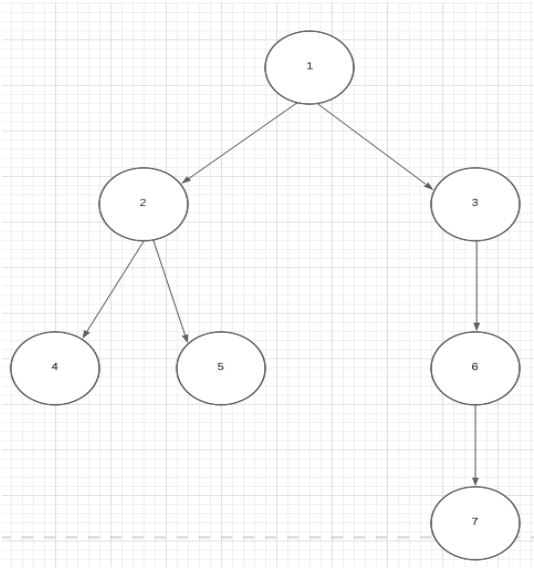
3.2.2 Algoritmo A*

Este algoritmo se divide en 2 partes. La primera de ellas es g(n), la cual representa el costo de la ruta desde su origen hasta un nodo n; h(n) representa el costo estimado de la ruta desde el nodo n al nodo de destino, la idea es lograr equilibrar g(n) y h(n) mientras se recorre, logrando así que f(n)=g(n)+h(n) tenga el costo más bajo posible. la complejidad de este algoritmo es $|h(x) - h^*(x)| = O(\log h^*(x))$ [6]



3.2.3 Algoritmo BFS

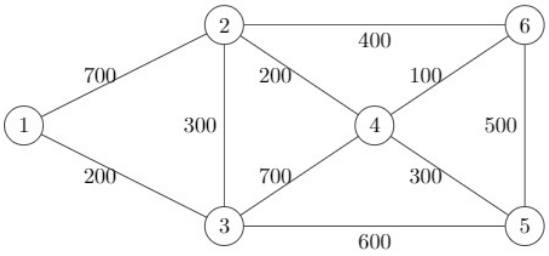
Breadth-First Search o BFS es un algoritmo que divide sus nodos en ramas, y todo empieza en la raíz la cuál desprende dichos nodos. Este algoritmo va recorriendo filas hasta llegar a su destino final, utiliza la estructura de datos de queue o cola para guardar sus vértices. La cola sigue el principio de First in First Out (FIFO), lo que hará que mostrará los vecinos del nodo, empezando por el nodo raíz. Este algoritmo tiene una complejidad de O(V+E), donde V es el número total de vértices y E es el número total de aristas. hay que tener presente que O(E) puede variar entre O(1) y $O(V^2)$ todo dependerá de qué tan denso sea el grafo[7]



3.2.4 Algoritmo de Floyd-Warshall

Este algoritmo trata de un análisis de grafos, para encontrar el camino mínimo en estos. En una ejecución, el algoritmo encuentra el recorrido entre todos los vértices. Dentro de este algoritmo, se crea una matriz Y de tamaño n*n y forma [i],[j], donde n son los vértices. Luego se crea una nueva matriz X dentro de de la ya existente Y, y es en X donde se

van a acabar encontrando los caminos que se le piden, ya que se va a reemplazar bien sea [i] o [j] por k, que va a ser el primer vértice que nos ayudará para encontrar el camino más corto. Este algoritmo tiene una complejidad de n^3. [8]



4. DISEÑO E IMPLEMENTACIÓN DEL ALGORITMO

A continuación, explicamos las estructuras de datos y los algoritmos utilizados en este trabajo. Las implementaciones de las estructuras de datos y los algoritmos están disponibles en Github[4].

4.1 Estructuras de datos

Lista de adyacencia utilizando diccionarios

(La estructura de los datos se presenta en la Figura 2.)

En lista de adyacencia se toma un vértice inicial y a partir de este se crea la lista enlazada con todos sus vértices adyacentes, esto gracias a la utilización de un diccionario que permite el acceso a estos valores. Dichos valores se guardan en una llave, en la cual los nodos adyacentes a estos se van agregando en caso de que el vértice de origen tenga una relación directa con otro vértice dentro del grafo, y va haciendo este proceso con cada uno de los vértices que se encuentren en el grafo. Los diccionarios son los encargados de guardar todas las llaves, y tener de una manera organizada el grafo elegido en forma de listas de adyacencia. Decidimos elegir las listas de adyacencia debido a que estas son bastante sencillas para representar aunque no siempre son las más eficientes en temas de espacio en memoria.

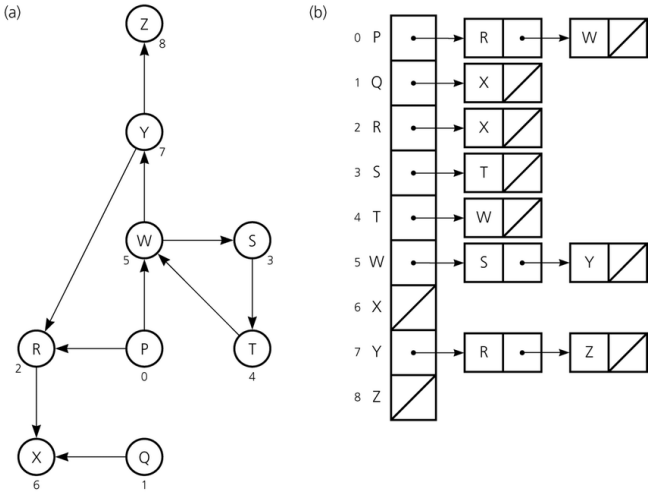


Figura 2: Un ejemplo de mapa de calles se presenta en (a) y su representación como lista de adyacencia en (b).

4.2 Algoritmos

En este trabajo, proponemos un algoritmo para un camino que minimiza tanto la distancia como el riesgo de acoso sexual callejero.

4.2.1 Algoritmo para un camino que reduce tanto la distancia como el riesgo de acoso sexual callejero

(El algoritmo se ejemplifica en la Figura 3.)

El algoritmo que implementamos fue el de Dijkstra, los nodos son cada una de las calles que hay en el mapa y las aristas tienen un peso, en este caso decidimos que ese peso se diera por multiplicación de distancia y riesgo + 0,1, este aumento de 0,1 en el riesgo se hace para cuando el riesgo es igual a 0 la multiplicación no de 0 y así tomar en cuenta la distancia también, de esta forma el algoritmo de Dijkstra recorrerá la ruta por donde los pesos sean menores, esto significa que para este camino va a hallar la ruta que reduzca tanto la distancia como el riesgo, dando como resultado el camino que nos es más útil para nuestra problemática.

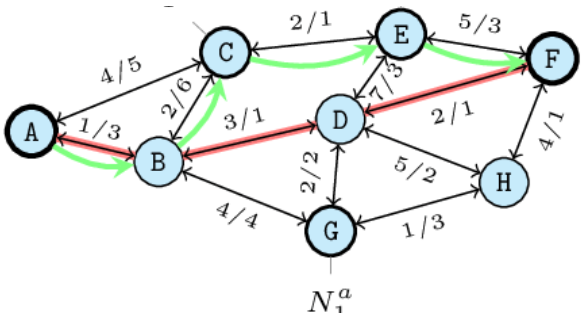


Figura 3: Cálculo de un camino que reduce tanto la distancia como el riesgo de acoso.

4.2.2 Cálculo de otros dos caminos para reducir tanto la distancia como el riesgo de acoso sexual callejro

[4] https://github.com/Facuddp/ST0245-002

A diferencia de la combinación mencionada anteriormente, la cual calculaba la multiplicación de la distancia por el riego + 0.1, en la estos dos caminos no tomamos las dos variables para el cálculo, o sea, en el camino que buscamos priorizar en la distancia, le damos al peso de la arista, el valor de la variable de distancia, mientras que en el último camino, en el cual, buscamos reducir el riesgo de la mayor forma, este valor será el de la variable del riesgo, donde dará como resultado un camino con el menor riesgo, sin combinar ambas variables en el cálculo de los caminos, consideramos que estos dos caminos no son los más ideales, pero son importantes según las necesidades del usuario, donde tendrá la opción de elegir un camino corto, un camino seguro, o uno que busque un nivel intermedio entre estos dos.

El algoritmo se ejemplifica en la Figura 4.

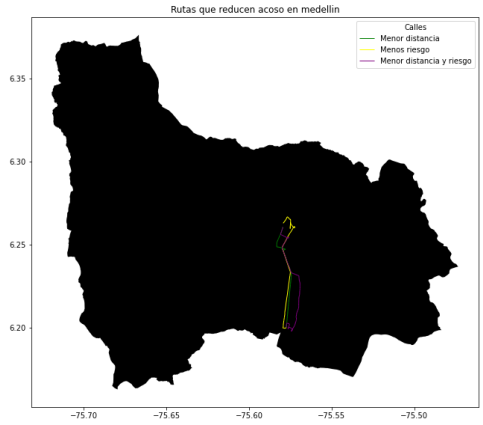


Figura 4: Mapa de la ciudad de Medellín donde se presentan tres caminos para peatones que reducen tanto el riesgo de acoso sexual como la distancia en metros entre la Universidad EAFIT y la Universidad Nacional.

4.3 Análisis de la complejidad del algoritmo

En el caso de Dijkstra, la complejidad para el peor caso se calcula de la siguiente manera: O(V) es el tiempo requerido para procesar el vértice; O(V) también es el tiempo requerido para llegar a todos los vértices del grafo. Si bien es cierto que en las listas de adyacencia la complejidad en memoria habitual es O(V+E), debido a que nada más requiere visitar al siguiente vértice, la complejidad de memoria es O(V), por lo que al final la complejidad pasa a ser de O(V)*O(V) = $O(V^2)$ para recorrer y procesar todos los datos requeridos para el funcionamiento del algoritmo.

Algoritmo	Complejidad temporal
Dijkstra	O(LogV (V+E))

Tabla 1: Complejidad temporal del algoritmo Dijkstra, donde V significa el número de vértices que contiene el grafo. Al inicio, todos los vértices están marcados como “no visitados”, lo que hace el algoritmo es iniciar el recorrido de los vértices y creando conexiones, las cuales son las E’s, estas “E” son las aristas que hacen las conexiones entre dichos vértices que se encuentran adyacentes entre sí, lo que hace que se creen estos caminos.

Estructura de datos	Complejidad de la memoria
Lista de adyacencia	O(V+E)

Tabla 2: Complejidad de memoria de las listas de adyacencia es O(V+E). V son los vértices de la lista mientras E son las aristas de la misma, las cuales hacen de conexión entre los vértices.

4.4 Criterios de diseño del algoritmo

Para la creación de nuestro proyecto tomamos en cuenta la complejidad, tanto del algoritmo como de la estructura de datos a utilizar. En nuestro caso, al implementar priority queues y el uso de las listas de adyacencia, la complejidad del algoritmo se redujo considerablemente, llegando a ser de O(LogV (V+E)). A su vez, vimos que en Dijsktra, el peor de los casos era de $O(V^2)$, si bien es cierto es un tiempo alto, era uno de los más eficientes para el problema que se nos presentaba en la búsqueda de caminos, ya que otros algoritmos que se podían usar para el problema, como el caso de A*, el cual tiene una notación de $O(V^m)$ en el peor de los casos, por lo que vimos al algoritmo Dijkstra como el más recomendable para afrontar la situación planteada. Algo similar ocurre cuando decidimos usar las listas de adyacencia para ser nuestra estructura de datos, ya que la complejidad en memoria de dicha estructura se acopla mejor con Dijkstra. Otra opción que teníamos era usar una matriz de adyacencia, pero que al final ocupaba más complejidad en memoria que las listas. Es por esta razón que decidimos usar tanto Dijkstra como las listas de adyacencia, ya que estás nos brindan un tiempo de ejecución corto y eficaz.

5. RESULTADOS

En esta sección, presentamos algunos resultados cuantitativos sobre los tres caminos que reducen tanto la distancia como el riesgo de acoso sexual callejero.

5.1 Resultados del camino que reduce tanto la distancia como el riesgo de acoso sexual callejero

A continuación, presentamos los resultados obtenidos de *tres caminos que reducen tanto la distancia como el acoso,* en la Tabla 3.

Origen	Destino	Distancia	Riesgo
Eafit	Unal	9700.925m	0.375
Eafit	Unal	7474.98m	0.342
Eafit	Unal	10128.88m	0.153

Tabla 3. Distancia en metros y riesgo de acoso sexual callejero (entre 0 y 1) para ir desde la Universidad EAFIT hasta la Universidad Nacional caminando.

5.2 Tiempos de ejecución del algoritmo

En la Tabla 4, explicamos la relación de los tiempos medios de ejecución de las consultas presentadas en la Tabla 3.

Calcule el tiempo de ejecución de las consultas presentadas en la Tabla 3. Indique los tiempos de ejecución medios.

Cálculo de v	Tiempos medios de ejecución (s)
v = distancia y riesgo	0.103 s
v = distancia	0.070 s
v = riesgo	0.047 s

Tabla 4: Tiempos de ejecución del nombre del *algoritmo Dijkstra* para cada uno de los tres caminos calculadores entre EAFIT y Universidad Nacional.

6. CONCLUSIONES

Como resultado obtenemos tres caminos diferentes para ir desde el origen al destino, en el primer camino combinamos el riego con la distancia, multiplicando estos dos, para los otros dos caminos tomamos en cuenta únicamente una variable, generando 3 camino diferentes totalmente, excepto cuando el camino es único o alguna calle sea al mismo tiempo la más segura y con menos distancia, ocasionando que los tres caminos pasen por la misma calle y provocará una ruta similar mas no igual, esto es útil al momento de querer tomar una ruta según la necesidad del usuario, dependiendo si quiere priorizar en la distancia, en la seguridad o algo promedio, desde nuestro punto de vista el mejor camino es el que toma ambas variables ya que, hay veces que el camino más corto no es el mejor y otras en el que el camino con menos riesgo se hace muy largo, por último, debido a que se pudo optimizar el código y tiene un tiempo de ejecución en pocos segundos, dándonos un tiempo de espera muy razonable para el usuario, debido a esto consideramos que es posible una implementación del proyecto en una situación real, donde el usuario segun sus deseos pdra consultar 3 rutas diferentes en poco tiempo

6.1 Trabajos futuros

En un futuro próximo queremos mejorar en la creación de una aplicación móvil, que también podrá ser puesta como una página web, ya que de esta manera más personas podrán aprovechar las funciones que brinda nuestro algoritmo, las cuales son bastante diferentes a las aplicaciones de GPS convencionales, gracias a que el usuario no solo podrá ver la distancia a la que se encuentra su destino, sino que también se suma la ruta más segura y un balance entre la distancia y el riesgo. Además, se podrían implementar más variables que le puedan ayudar al usuario a escoger una ruta según las preferencias que este tenga. Otra posibilidad es implementar un aprendizaje automático, este ayudará a que el funcionamiento de la aplicación sea aún más preciso, ya que recolectará las rutas usadas por los demás usuarios, con el fin de recomendar el camino más óptimo en los campos elegidos por el usuario.

AGRADECIMIENTOS

Los autores agradecen a Samuel Rico y Gregorio Bermúdez, monitores de Estructura de Datos y Algoritmos 1 de la Universidad EAFIT, por facilitar la explicación y creación del código utilizado para la creación de nuestro proyecto.

Los autores agradecen al profesor Juan Carlos Duque, de la Universidad EAFIT, por facilitar los datos de la Encuesta de Calidad de Vida de Medellín, de 2017, procesados en un archivo *Shapefile*.

REFERENCIAS

1. Bach. M.4 2015. Desarrollo de un sistema para la optimización de rutas de trabajo utilizando el algoritmo de Dijkstra y diagramas de Voronoi https://idoc.pub/documents/proyecto-de-tesis-5s-wl1p082kj9lj.
2. Rodriguez, L. 2005. Algoritmos para calcular la ruta más corta en la malla vial de la ciudad de Bogotá. Bogotá. https://repositorio.uniandes.edu.co/bitstream/handle/1992/22379/u263427.pdf
3. Handy Permana, S., Arifitama, B., Bintoro, K. and Syahputra, A. *Comparative Analysis of Pathfinding Algorithms A *, Dijkstra, and BFS on Maze Runner Game*, 2018. https://www.researchgate.net/profile/Budi-Arifitama/publication/325368698_Comparative_Analysis_of_Pathfinding_Algorithms_A_Dijkstra_and_BFS_on_Maze_Runner_Game/links/5b36489d4585150

d23e1d802/Comparative-Analysis-of-Pathfinding-Algorithms-A-Dijkstra-and-BFS-on-Maze-Runner-Game.pdf.

4. 1.Cardona, M., Tinoco, H. and Castrilón, O. *Determinación del Método Óptimo de Operaciones de Ensamble Bimanual con el Algoritmo de Dijkstra (o de Caminos Mínimos)*, 2017. https://www.scielo.cl/pdf/infotec/v28n4/art15.pdf.
5. Algoritmo de Dijkstra - EcuRed. *Ecured.cu*. https://www.ecured.cu/Algoritmo_de_Dijkstra.
6. Algoritmo A * - Algoritmos de grafos - Detección de rutas más cortas. *GraphEverywhere*. https://www.grapheverywhere.com/algoritmo-a/.
7. Búsqueda en amplitud (BFS): implementación iterativa y recursiva. *Techiedelight.com*. https://www.techiedelight.com/es/breadth-first-search/.
8. Algoritmo de Floyd-Warshall.. Medium. https://medium.com/algoritmo-floyd-warshall/algoritmo-de-floyd-warshall-e1fd1a900d8.