RUTA ÓPTIMA PARA LA MOVILIDAD Y SEGURIDAD CIUDADANA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Miguel Ángel Escudero C.  Universidad Eafit  Colombia  maescuderc@eafit.edu.co | Luis Felipe Yánez S.  Universidad Eafit  Colombia  lfyanezs@eafit.edu.co | Mauricio Toro  Universidad Eafit  Colombia  mtorobe@eafit.edu.co | Andrea Serna Universidad Eafit Colombia asernac1@eafit.edu.co | Mauricio Toro  Universidad Eafit  Colombimtorobe@eafit.edu.co |

# **RESUMEN**

El objetivo del proyecto es encontrar una ruta optima tomando en consideración los factores de distancia del recorrido y un riego de acoso callejero. Esto con el objetivo de reducir los índices de inseguridad y desconfianza de los ciudadanos, además, de brindar apoyo a la movilidad y así aumentar la eficiencia en las vías.

## **Palabras clave**

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

# **1. INTRODUCCIÓN**

Con el aumento de la inseguridad que se ve generalizada en los últimos años, también se ha aumentado los niveles de acoso callejero, esto se ve reflejado en que según cifras de la secretaria de las mujeres de la alcaldía el 60% de las mujeres dijeron sentir que Medellín no es una ciudad segura para ellas, por esta razón, se quiere brindar apoyo principalmente a la población más afectada pero también al público en general para así evitar que suceda esta reprochable situación y hacer más cómoda la movilización en la ciudad.

# **1.1. Problema**

Teniendo como factores principales las problemáticas de movilidad y de seguridad en específico acoso callejero, los cuales tienen un gran impacto generalizado en la inseguridad y en la calidad de vida de los habitantes, se plantea la solución mediante un algoritmo que nos ayude a medir de manera óptima nuestra ruta de movilización, para así evitar contratiempos y también elegir una ruta más segura.

**1.3 Estructura del artículo**

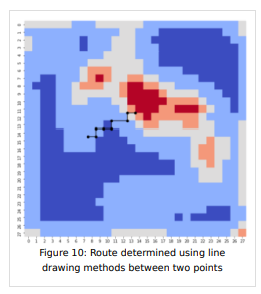
A continuación, en la Sección 2, presentamos trabajos relacionados con el problema. Posteriormente, en la Sección 3, presentamos los conjuntos de datos y los métodos utilizados en esta investigación. En la Sección 4, presentamos el diseño del algoritmo. Después, en la Sección 5, presentamos los resultados.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

## **3.1 Previniendo acoso sexual utilizando un algoritmo de búsqueda de caminos usando Nearby Search**

Este busca reducir el acoso sexual callejero como modelo utilizaron la ciudad de Mumbai, el algoritmo con nombre “Bresenham’s line algorithm” el cual determina dos puntos(E,NE) de un cuadro “grip” y compara cruzando a través de estos una línea (M) para determinar su nivel de riesgo, esta acción la repite una y otra vez hasta conseguir llegar al punto de llegada, como resultado nos da el siguiente mapa



Ma, D., 2020, “Preventing Sexual Harassment Through a Path Finding Algorithm Using Nearby Search” .

## **3.2 Sistema de análisis e integración de datos para la planeación de una ruta segura**

Este busca reducir los crímenes contra la mujer en la india ,en este caso se utilizaron dos algoritmos, el primero para hallar el valor de seguridad y se modificó el “Dijsktra’a algorithm” para utilizar el valor de seguridad encontrado en el primer algoritmo como una condición para hallar la ruta más optima , aún faltan evaluación de resultados probatorios para determinar que el algoritmo funcione de la manera esperada ,algunos resultados serian, el aumento de los ciclistas, peatones y la seguridad de las mujeres en las calles.

Sarraf, R., McGuire, M., 2018, “A data integration and analysis system for safe route planning”.

## **3.3 ruta segura: un modelo robusto para la predicción de rutas seguras usando los índices de delincuencia y accidentalidad**

Enfocado hacia evitar zonas peligrosas en base a los datos de accidentalidad y de arrestos de la zona para así evitar pasar por lugares con una clasificación peligrosa ,algoritmo que utilizaron, el algoritmo es llamado “Mask algorithm” el cual mide los parámetros de peligrosidad en base a los datos de arrestos y accidentes de la “NYPD” donde a través de “clusters” integra tres algoritmos los cuales miden zona peligrosa, tu ubicación y usuarios cercanos , como resultado tenemos una zona fraccionada por “clusters” los cuales cuentan con una puntuación que entre menor más segura es para transitar, en base a cada uno de estos puntos se puede checar manualmente la puntuación de cada “cluster” y elegir la mejor ruta.

Soni, S., Gauri, V., Chaurasia, S., Chaurasia, D.2019 “Route.The Safe: A Robust Model for Safest Route Prediction Using Crime and Accidental data”

## **3.4 Un creador de rutas para turistas consciente a la seguridad basado en open data y VGI**

Este busca mejorar la experiencia de los conductores en las vías, hacerlo de manera personalizada, el algoritmo integra restricción de rutas por peligrosidad, también una “cost surface”, se utilizó el “IDW interpolation method” que lo que hace es que cada valor tiene un peso y tiene una distancia inversamente proporcional que tiene el algoritmo a los valores aproximados del punto, además de usr el “OPTICS algorithm”. Los resultados hallados en esta prueba fueron 3 figuras en las cuales al comparar se vieron diferencias entre la ruta más rápida y la más segura, además se noto un cambio al agregar “obstacle polygons” como una nueva restricción se pudo observar un cambio aún más optimo

Keler, A., Damascene, J. (2016) “Safety-aware routing for motorized tourist based on open data and VGI”, journal of Location Based Services, 10:1, 64-77

## **3. MATERIALES Y MÉTODOS**

En esta sección, explicamos cómo se recogieron y procesaron los datos y, después, diferentes alternativas de algoritmos del camino más corto restringido para abordar el acoso sexual callejero.

## **3.1 Recogida y tratamiento de datos**

El mapa de Medellín se obtuvo de Open Street Maps (OSM) [[1]](#footnote-1)y se descargó utilizando la API[[2]](#footnote-2) OSMnx de Python. La (i) 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 se obtuvieron de los metadatos proporcionados por OSM.

Para este proyecto, se calculó la 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 normalizó, 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]](#footnote-3).

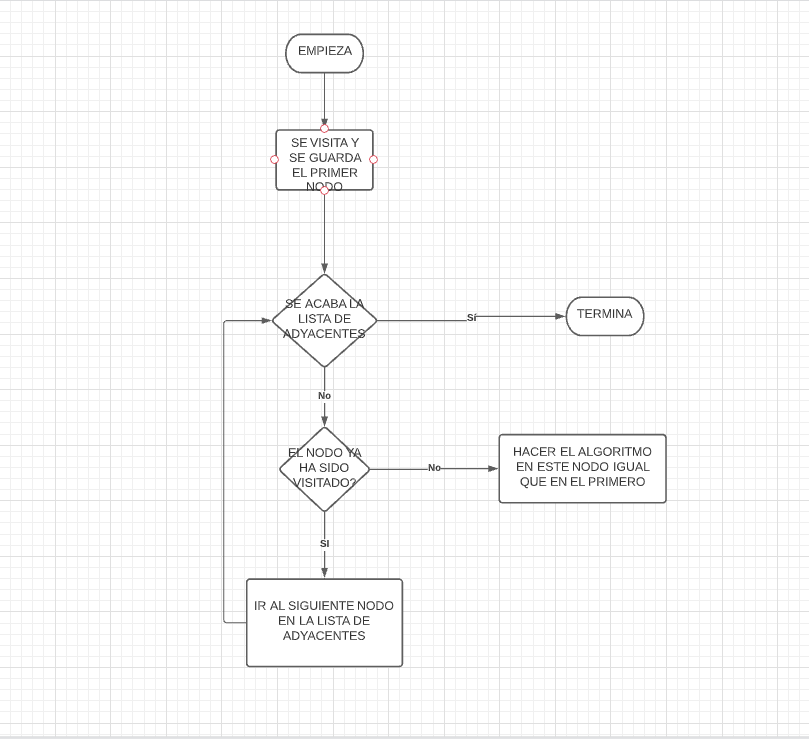
**Figura 1.** Riesgo de acoso sexual calculado como una combinación lineal de la fracción de hogares que se sienten inseguros y la fracción de hogares con ingresos inferiores a un salario mínimo, obtenida de la Encuesta de Calidad de Vida de Medellín, de 2017.

## **3.2 Alternativas de camino más corto con restricciones**

## A continuación, presentamos diferentes algoritmos utilizados para el camino más corto restringido. *(En este semestre, ejemplos de dichos algoritmos son DFS, BFS, una versión modificada de Dijkstra, una versión modificada de A\*, entre otros).*

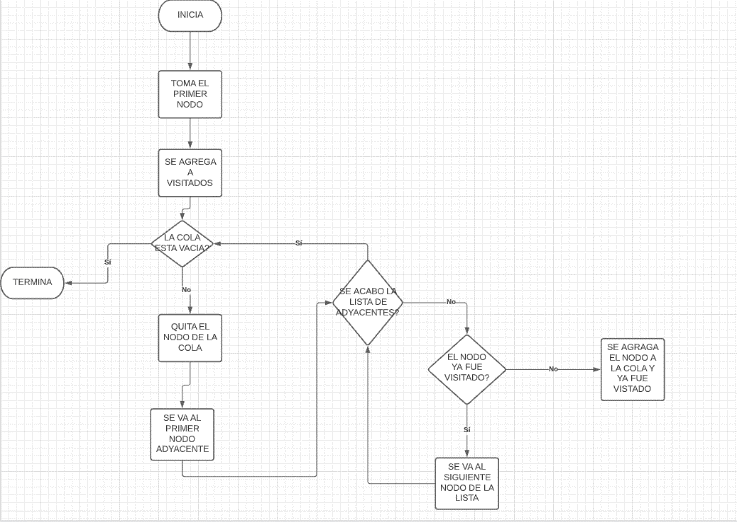
**3.2.1 DFS (Depth First Search)**

El algoritmo DFS lo que hace en simples palabras es agarrar un vértice de un gráfico y ponerlo al principio de un stack y se agrega a una lista de “visitados” y después se hace otra lista de los nodos adyacentes al primero y se busca cuales no están en la lista de visitados y se agregan al principio del stack los nodos adyacentes encontrados y esto se repite hasta que no hay nodos que no hayan sido visitados.



Referencia: https://www.programiz.com/dsa/graph-dfs

**3.2.2 Breadth First Search (BFS)**

El algoritmo BFS en simples palabras lo que hace es agarrar cualquier vértice de un gráfico en lo último de una cola/queue y después se coge el primer ítem de la cola y se añade a la lista de visitados, después se crea una lista de los nodos adyacentes a ese vértice y se agregan al final de la cola los que no estén en la lista de visitados, este proceso se repite hasta que la cola se encuentre vacía. 

Referencia: https://www.programiz.com/dsa/graph-bfs

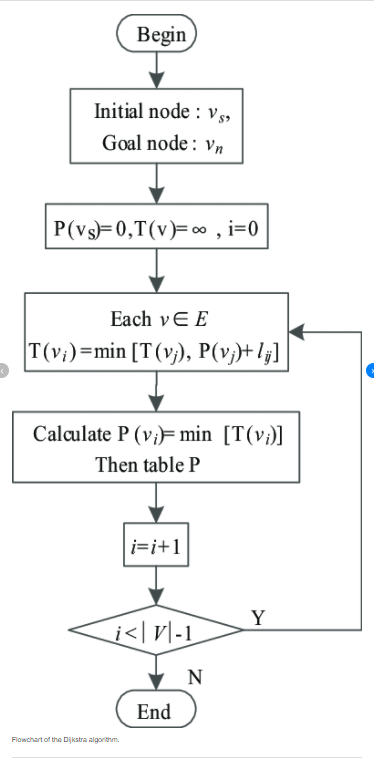
**3.2.3 Backtracking**

Se tienen 4 posibles caminos para recorrer la matriz siendo estos arriba (x, y-1), abajo (x, y+1), derecha (x+1, y), izquierda (x-1, y), y el algoritmo toma un nodo y al revisar si se puede hacer al menos una de estos caminos y llegar al destino se guarda el tamaño del camino recorrido, si no es válido ninguno de los 4 caminos entonces se busca otro nodo y se repite el mismo proceso, y al final el algoritmo escoge el camino con menor

Referencia: 4 elemento en el github de mauricio

**3.2.4 Dijkstra’s**

Se escoge el vértice inicial y se le asignan infinitos valores de caminos a los otros nodos, se marca el vértice inicial como visitado y después se actualiza la distacia de los nodos vecinos y se escoge cual representa la menor longitud, al de menor longitud se marca como visitado y forma parte del camino más corto, se agregan al camino más corto los valores más cercanos cada que se actualizan los valores, después se actualiza los valores de los nodos adyacentes al valor que este recién añadido al camino más corto



Referencia: <https://www.freecodecamp.org/news/dijkstras-shortest-path-algorithm-visual-introduction/>

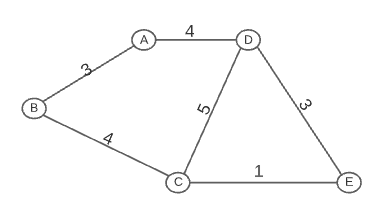
Imagen: https://www.researchgate.net/figure/Flowchart-of-the-Dijkstra-algorithm\_fig1\_327982778

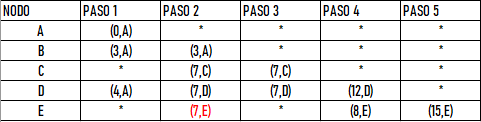
## **4. DISEÑO E IMPLEMENTACIÓN DEL ALGORITMO**

## 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]](#footnote-4).

## La estructura de datos que se utilizó fue una lista de adyacencia dado que utilizamos el algoritmo Dijkstra

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





Utilizamos el algoritmo Dijkstra para implementar la solución de nuestro problema e hicimos una gráfica la cual consiste en ir de nodo en nodo progresivamente sumando las distancias para hallar el camino más corto entre A y E en este caso específico, lo cual nos dio una distancia de 7 por el camino más optimo.

**4.2 Algoritmos**

En este trabajo, proponemos algoritmos para el problema del camino más corto restringido. El primer algoritmo calcula el camino más corto sin superar un riesgo medio ponderado de acoso *r*. El segundo algoritmo calcula el camino con el menor riesgo medio ponderado de acoso sin superar una distancia *d*.

**4.2.1 Primer algoritmo**

El algoritmo que implementamos nosotros fue el dijsktra , el cual utilizamos en Python las librerías numpy, pandas y networkx. Con pd.read\_csv leemos la información del mapa de medellin y le definimos con la ayuda de la librería networkx el valor de origen, destino y a longitud final, finalmente, devuelve el valor con nx.dijsktra.path.

El algoritmo se ejemplifica en la Figura 3.



# **REFERENCIAS**

1.Daniel Ma, Preventing Sexual Harassment Through a Path Finding Algorithm Using Nearby Search, omeda, 2020

2. Sarraf, R., McGuire, M., “A data integration and analysis system for safe route planning”,2018.

3.Soni, S., Gauri, V., Chaurasia, S., Chaurasia, D. “Route-The Safe: A Robust Model for Safest Route Prediction Using Crime and Accidental data”2019.

4. Keler, A., Damascene, J. (2016) “Safety-aware routing for motorized tourist based on open data and VGI”, journal of Location Based Services, 10:1, 64-77

5. Programiz, https://www.programiz.com/dsa/graph-dfs

6. Programiz <https://www.programiz.com/dsa/graph-bfs>

7. pencilprogrames, <http://pencilprogrammer.com>

8. Cassingena, E. <https://www.freecodecamp.org/news/dijkstras-shortest-path-algorithm-visual-introduction/>

1. <https://www.openstreetmap.org/> [↑](#footnote-ref-1)
2. https://osmnx.readthedocs.io/ [↑](#footnote-ref-2)
3. [https://github.com/mauriciotoro/ST0245Eafit/tree/master/  
   proyecto/Datasets/](https://github.com/mauriciotoro/ST0245Eafit/tree/master/proyecto/Datasets)  [↑](#footnote-ref-3)
4. http://www.github.com/ ????????? /.../proyecto/ [↑](#footnote-ref-4)