ALGORITMOS QUE REDUCEN DISTANCIAS Y EL ACOSO CALLEJERO PARA PEATONES EN MEDELLÍN

Sofía Jaramillo
Universidad Eafit
Colombia
sjaramil42@eafit.edu.co

Jerónimo Guerrero
Universidad Eafit
Colombia
jguerrero2@eafit.edu.co

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

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

RESUMEN

El acoso callejero como un problema social y cultural que afecta en mayor medida a los peatones de ciertas rutas. Esta es una de las formas de violencia más comunes, pero debido a factores culturales, se ha convertido en conductas naturalizadas que se ven en las calles y son poco tratadas, por lo que han adquirido cierto grado de aceptación social. Así pues, los peatones tienen más riesgo al pasar por algunas rutas en donde los índices de acoso callejero son más altos debido a la poca atención que tiene este problema en la sociedad. Se propone implementar el algoritmo de Dijkstra para encontrar el camino con el menor peso, que en este caso son 3 caminos con 3 variables distintas, en donde la primera variable v=d*r tuvo un tiempo de ejecución de 0,065 segundos, v=d/100 + r 0,06 segundos y v=d+10*r 0,07 segundos. En donde la variable v=d/100+r es recomendada debido a la distancia recorrida de 8080,76 metros y su riesgo promedio de 0,55. En donde el tiempo de ejecución que es óptimo de 0,06 segundos para encontrar el camino con menor peso.

Palabras clave

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

1. INTRODUCCIÓN

Aplicaciones como Google Maps, Waze y otras, no consideran el acoso en su búsqueda de rutas, por lo que existe la necesidad de encontrar algoritmos para reducir la distancia de las rutas teniendo en cuenta los niveles de acoso callejero. Para recomendar rutas, se tiene en cuenta la optimización de las distancias, pero, por otro lado, hay que tener en cuenta que las personas se sienten más seguras al pasar por algunos lugares ya que en estos se presentan menos casos de violencia. Entonces, se crea la necesidad de encontrar algoritmos que consideren al mismo tiempo, la seguridad y la distancia entre rutas. Estos algoritmos serán implementados para las calles de Medellín. En la ciudad hay zonas mayormente afectadas por el acoso callejero, como lo es el centro de Medellín, que es uno de los principales sectores de la ciudad afectados por esta problemática debido al alto número de delitos sexuales y violencia callejera en esta zona.

1.1. Problema

El acoso callejero vulnera la integridad de las personas, sobre todo mujeres, que conlleva a traumas psicológicos. En donde a su vez, la normalización de este, hace parte de una construcción social basada en estereotipos, especialmente sexistas, en donde las personas consideran el acoso como parte de acciones cotidianas y se tiene una menor percepción de su gravedad, esto gracias a una sociedad que sigue arraigada por el machismo. Por lo que al transitar por algunas calles se incrementa el riesgo de ser acosado. De esta manera, es necesario encontrar alternativas para la reducción y prevención del acoso callejero por medio de la creación de algoritmos para la búsqueda de rutas, y así lograr que los peatones transiten por zonas más seguras que tengan menos niveles de violencia, pero que al mismo tiempo se reduzcan las distancias.

1.2 Solución

Los datos recopilados llegan a servir mucho para la prevención y tratamiento de un problema, en este caso, se tiene el problema de los altos niveles de acoso callejero debido a la inseguridad en las calles de Medellín, por ende, se propone una solución que se centra en la prevención del problema implementando un algoritmo de búsqueda que basado en los datos recopilados, encuentra el camino más seguro a comparación con los otros caminos en base al riesgo de acoso, y que al mismo tiempo priorice el camino más corto. Como en este caso tenemos un punto de origen y un objetivo de llegada que es el origen, con valores no negativos que serían el peso y el riesgo de abuso, podemos implementar el algoritmo de Dijkstra, ya que es un algoritmo de búsqueda eficiente, que permite desplazarse desde un nodo inicial hacía los otros nodos, visitando todos

los nodos hasta el nodo final encontrando el menor peso para llegar a la solución óptima.

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.

2.1 Rutas Seguras: Un modelo para predecir la ruta más segura usando datos de delitos y accidentes.

En el trabajo realizado, presentan un método innovador para encontrar la ruta más segura en donde se corra el menor riesgo. Los crímenes están aumentando día tras día y esto preocupa cada vez más a las personas, ya que la seguridad se ve como una prioridad. Hay algunas personas que no tienen conocimiento de cuáles rutas tomar, por lo que usan google maps, pero este solo da el camino más corto y no considera la ruta más segura.

La solución que se propone se centra en: predecir rutas seguras usando los datos de crímenes y accidentes, pero a su vez, considerando la distancia entre la locación actual y el destino. Se usan los datos disponibles en NYC OpenData para determinar un aproximado de riesgo.

Se realizó a partir de algoritmos de Machine learning para generar el camino más seguro basado en los valores aproximados de riesgo de los sectores. Se dividió a NYC en pequeñas regiones para realizar mejores predicciones. Además, se eliminaron los valores null y nan, y se analizaron los outliers(datos extraños) que al final se descartaron.

Se usó el “Mask Algorithm”, después se modeló el diseño del algoritmo teniendo en cuenta la cantidad de accidentes y crímenes obtenidos en los datos que se agruparon por zonas(K), después del etiquetado de variables y plantear los modelos el paso final es encontrar la ruta más segura.

Después de obtener los resultados de K(agrupamiento) en NYC, se predijo la criminalidad y accidentalidad usando R2 score, y se determinó el coeficiente de regresión para verificar la validez de los datos. Basado en estas predicciones se busca la ruta más segura que involucra dos condiciones, si hay una sola ruta que tiene el riesgo más bajo, esa es la ruta más segura. Si no, si hay más de una ruta con el menor riesgo, entonces se comparan las distancias, y la ruta con la menor distancia se considera como la ruta más segura.

1. Shivangi Soni, Venkatesh Gauri Shankar, Sandeep Chaurasia. Route-The Safe: *A Robust Model for Safest Route Prediction Using Crime and Accidental Data*, School of Computing and IT, Manipal University Jaipan, India, Diciembre 2019.

2.2 Integración de datos y análisis del sistema para planificación de rutas seguras.

Los casos de abusos sexuales en contra de la mujer es el cuarto problema más grande en India que se ha incrementado exponencialmente en la última década. Se busca la manera de resolver el crecimiento de este número, diseñando una aplicación para rutas seguras y rápidas, en lugar de únicamente la ruta más rápida. Se propone encontrar las condiciones en las que suceden este tipo de casos, y prevenir estas condiciones pero, sobre todo, evitar lugares donde suceden estas condiciones. Se propone desarrollar un modelo que diga exactamente qué tan seguro es un lugar, tomando en cuenta los factores que pueden influenciar los abusos en el área, e incorporar los niveles de seguridad en un mapa. Todo esto, para reducir el número de crímenes contra las mujeres en India.

Se determinan los factores que hacen que una zona se considere o no segura y la seguridad total es la suma de estos factores. Se grafican las funciones considerando cada factor y viendo su cambio, planteando el algoritmo para encontrar el mecanismo para encontrar la ruta más práctica.

Se implementó el “Dijkstra´s Algorithm” para encontrar el camino más corto entre nodos en un grafo, va desde un nodo hasta el destino y se detiene cuando ha determinado el patrón más corto desde el nodo hasta el destino.

Aunque se creen algoritmos para promover las rutas seguras para mujeres, una de las mayores limitaciones es la cantidad de datos que limita calcular el nivel de seguridad. Por lo que se tomaron en cuenta los factores que determinan una zona como segura. La ruta de seguridad puede llegar a sugerir un

patrón seguro pero que es muy largo, en un futuro se espera implementar preguntas para que el usuario seleccione la ruta, realizando un algoritmo de decisión que analice las respuestas y genere la mejor ruta para el usuario.

1. Aryan Guptaa, Bhavye Khetan. *A Data Integration and Analysis System for Safe Route Planning*, International Journal of Science and Research, 2018.

2.3 Seguridad vial, a un paso de la muerte

En los últimos años uno de los índices más altos de muertes es por accidentes de tránsitos, ubicándose de terceros, por debajo del cáncer y problemas del corazón, donde además hay unos factores que no solo dependen de los carros, hay factores externos como la vía, ambientales, peatones, entre otros.

Al observar este número de fallecidos o de heridos, se empezó a crear aquellas ayudas para conductores, para poder evitar accidentes, donde por medio de aplicaciones nos muestran toda la información posible de la vía y el entorno, para poder facilitar la vida de los conductores, debido a que al mencionar, previene frente a cualquier situación al que va al volante, ya sea una simple cámara de seguridad, o como se encuentra la calle, o si hay tráfico, cualquier factor ajeno al carro.

Se puede ver un modelado de colisión de vehículos en la intersección (LA = largo del vehículo A sujeto; LB = largo del otro vehículo B; WA = ancho del vehículo A sujeto; WB = ancho del otro vehículo B), donde la mayoría de los accidentes surgen cuando el carro B (el que se encuentra detrás) no alcanza a frenar por un frenado en seco del carro A (el que está adelante).

Sumándole a eso que la reacción para una persona >= 51 años es de 2 segundos, mientras que < 51 es de 1 segundo, que es un gran tiempo para estar atento frente a una situación extraordinaria.

Aquellos algoritmos que se usan son sobre todo de física, para poder calcular aquellas velocidades, masa del vehículo, tiempo de frenado, que sirve para tener claro todo aquello para poder tener un buen viaje sin ningún riesgo.

Se ha evidenciado un gran cambio a la hora de accidentes, debido a que apps que digan y adviertan toda esta información pone en modo alerta al conductor, como cuando hay un choque, o hay un cierre de la vía, cualquier circunstancia, esto salva vidas y genera un gran cambio en la sociedad.

1. Zhaoxiang He, Xiao Qin. *Incorporating a Safety Index into Pathfinding*, 2022.

2.4 Más allá de una corta ruta, primero la calidad de ella.

En el artículo trabajado exponen una problemática que día a día afecta a millones de personas, la seguridad en las calles de la ciudad, debido a que en estos momentos lo único que importa o importaba era llegar más rápido a un destino, pero al empezar a ver tantos casos de inseguridad, hurto, a peatones cambió esta prioridad.

Se empezó a generar nuevas ideas para solucionar aquellas situaciones, en estos momentos ya hay algunas aplicaciones que muestran no solo el camino más rápido, sino el más seguro, que tenga un índice de violencia, accidentes más bajos, sin embargo reconoce más cosas, como la mejor ruta para personas en silla de ruedas, o rutas donde se pase por lugares de comercio o restaurantes, ya hay una gran base de datos, donde se va acumulando toda ésta información, además para reconocer si es o no segura esa ruta, ingresan y buscan si han habido casos recientemente y que porcentaje tiene, revisando noticias, redes sociales, entre muchas otros recursos.

Para esto se usó una gran fuente de datos para poder mostrar toda la información ya mencionada anteriormente, OpenStreetMap (OSM), una de las bases de datos geográficas más grandes, funciona con nodos, y para conseguir aquella información se hace con una interfaz, OSM usa direcciones como líneas y ya lugares específicos que se toman como puntos.

Google Places, otra base de datos, pero con información más específica, es decir, este programa sirve sobre todo a las personas en situación de discapacidad, debido a que muestra por medio de fotos como son los andenes, el barrio o aquel lugar al que quiera ir la persona.

Algoritmos usados para el enrutamiento: algoritmo de “Dijkstra”, consta de un nodo inicial j, en un gráfico k, esto lo que hace es buscar el camino más corto entre los puntos j y el punto de destino f, hay otro algoritmo que busca todo lo que ha pasado en las calles para evaluar si es segura o no.

Se ha visualizado un gran cambio en estos últimos tiempos, ya no solo importa la distancia en tiempo, o si es más corto o largo el camino, sino que ya los usuarios que utilizan frecuentemente todas las aplicaciones se preocupan más por el bienestar y seguridad.

Panote Siriaraya, Yuanyuan Wang, Yihong Zhang, Shoko Wakayamiya, Péter Jeszenszky, Yukiko Kawai, Adam Jatowt. *Beyond the Shortest Route: A Survey on Quality-Aware Route Navigation for Pedestrians*, Kyoto Institute of Technology, Yamaguchi University, Osaka University, Nara Institute of Science and Technology, Ritsumeikan University, 2022.

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].

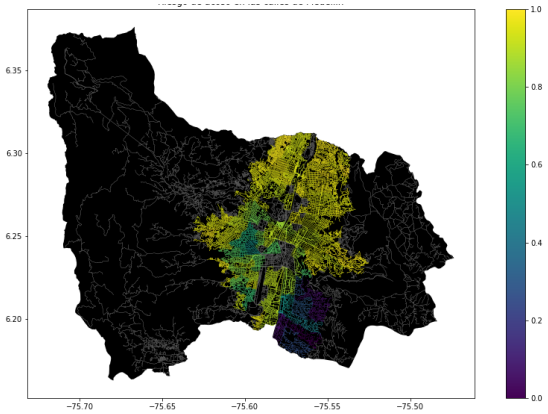


Figura 1. Riesgo de acoso sexual calculado como una combinación lineal de la fracción de hogares que se sienten inseguros y la fracción de hogares con ingresos inferiores a un salario mínimo, obtenidas de la Encuesta de Calidad de Vida de Medellín, de 2017.

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.

3.2.1 DFS

Depth first search (Algoritmo de búsqueda en profundidad)

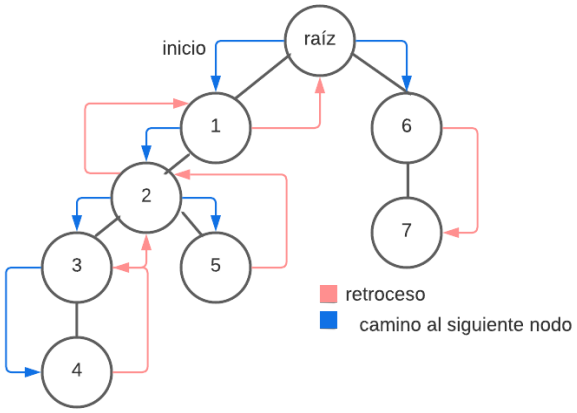
Es un algoritmo de búsqueda para encontrar todos los vértices alcanzables de un grafo desde el vértice de origen, mientras recorren los nodos en un camino correcto. Cuando ya no quedan más nodos por recorrer, se devuelve al vértice y repite el proceso con un camino nuevo. Se puede definir desde la búsqueda de un nodo padre a un nodo hijo. Se usa cuando hay que probar si una solución entre varias posibles cumple con los requisitos.

Complejidad de DFS: O(V + E) donde V es el número total de vértices y E el número total de aristas en el grafo. La complejidad también depende de qué tan denso sea el grafo.

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

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

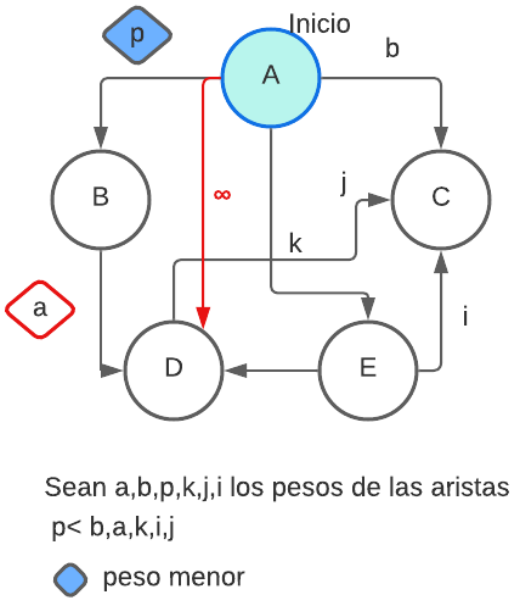
[3]https://github.com/mauriciotoro/ST0245Eafit/tree/master/proyecto/Datasets/



.3.2.2 Dijkstra

Algoritmo para encontrar el camino más corto, se clasifica como un algoritmo de búsqueda, es un método iterativo. Sirve para encontrar el camino mínimo desde un nodo origen a todos los demás nodos del grafo. Trabaja por etapas, y toma la mejor decisión, en donde el óptimo puede modificarse si se encuentra una solución mejor. Toma en cuenta los valores de las aristas. Los pesos de las aristas no pueden ser negativos para implementar este algoritmo. Utiliza el principio que dice que para que un camino sea óptimo, entonces todos sus caminos deben ser óptimos. Se busca el vértice que esté más cerca del punto de origen y se ve si se puede llegar más rápido por ese vértice, después se toma el siguiente más cercano y se repite el proceso. Esto hasta llegar al vértice que es el destino. Por ejemplo, si los vértices del grafo representan ciudades y el peso de las aristas sean las distancias entre dos ciudades, el algoritmo puede ser usado para determinar la ruta más corta de una ciudad a otra.

El algoritmo de Dijkstra es un algoritmo eficiente ya que su complejidad es de O(n^2), donde “n” es el número de vértices)

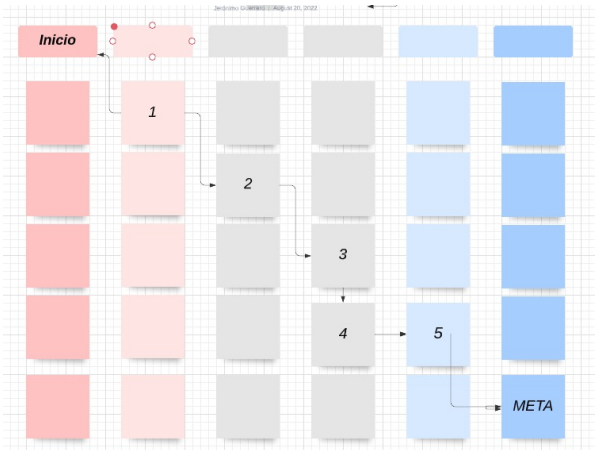


3.2.3 A *

Es un algoritmo que se usa para encontrar una ruta más óptima de un nodo de inicio a un nodo de meta, a través de un espacio de problema, sin embargo tiene un margen de error y a veces no usa la opción más óptima, además de que es algo más heurístico, y se saca de esta forma.

f(n) = g(n) + h(n) La función se encuentra compuesta por: g(n):es el costo de las movidas realizadas.

h(n) : es la función heurística. Representa el costo estimado del mejor camino.



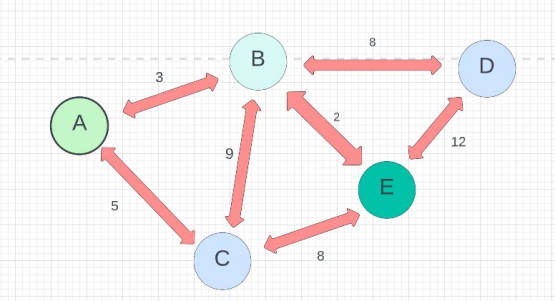
Se puede evidenciar que a pesar de que es un camino rápido, hay veces que no encuentra la forma óptima y por ende no es el más rápido.

En el caso peor, con una heurística de pésima calidad, la complejidad será exponencial(O(2^n))

3.2.4 Floyd

El algoritmo de Floyd es la opción utilizada cuando se desea determinar el camino mínimo entre todos los pares de vértices de un grafo, comparando todos los posibles caminos cuando logra mejorar la estimación para llegar una opción óptima.

La complejidad de este algoritmo es O(n^3). El algoritmo resuelve eficientemente la búsqueda de todos los caminos más cortos entre cualesquiera nodos



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

La estructura de Datos que utilizamos e implementamos para este proyecto fue la lista de adyacencia con diccionarios. Inicialmente utilizamos pandas para definir una nueva estructura de datos en un array de la librería numpy para leer y manipular archivos csv, el cual es el tipo de archivo en donde se encuentran todos los datos a utilizar.

Así mismo se empleó listas enlazadas**,** donde se habla de un nodo anterior, del actual y del siguiente, como su nombre lo indica, es una estructura que contiene elementos conectados mediante un enlace, las listas enlazadas están formados por un nodo, cada nodo contiene dos atributos, uno donde almacena los datos y el otro para enlazarse con el siguiente nodo.

Un diccionario en Python es una estructura para trabajar con colecciones de datos almacenados en pares de valores, una vez creas el diccionario puedes acceder a cada valor utilizando la clave.

La lista de adyacencia con diccionarios permite representar el grafo, para crear llaves con las variables origen y destino, los valores en el diccionario son las coordenadas de los nodos de adyacencia con su respectiva distancia y el nivel de abuso. La estructura de los datos se presenta en la Figura 2.

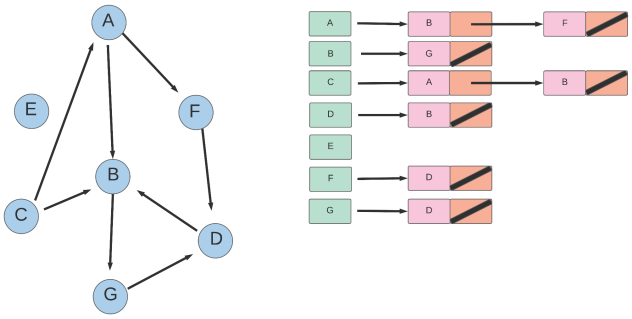


Figura 2: Estructura de lista de adyacencia aplicado en un grafo.

[4] http://www.github.com/ sofiajaramilloch /proyecto.git

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

Para encontrar el camino más corto y más seguro decidimos implementar el algoritmo de Dijkstra para encontrar la solución, ya que es un algoritmo eficiente que nos permite encontrar el menor de un nodo de origen a los otros nodos,

Para implementar el algoritmo de búsqueda, se usó un diccionario que representa la distancia desde un punto de origen a todos los nodos Al estar trabajando con Dijkstra todas las distancias **d** deben estar inicializadas con un valor infinito relativo (muy grande) en **D** ya que todas las distancias son desconocidas al principio. Exceptuando la del origen que es cero porque es nuestro nodo de partida, y la distancia del origen con respecto al origen es cero. Es decir, la distancia entre todos los nodos diferentes del origen con el origen es (infinito, -) Un valor muy grande sin nodo anterior porque no se sabe ninguna de los dos. Y el origen (0,-) porque la distancia es cero y no le precede ningún nodo ya que es el origen. Después de eso se selecciona un nodo actual **a** tal que inicialmente **a=x** que inicialmente es el origen. Luego, se recorren los nodos adyacentes a **a,** exceptuando los nodos visitados.

Si la distancia de **a** hasta el nodo adyacente no visitado guardada en **D** es mayor que la distancia de **x** hasta **a** más la suma de **a** hasta el nodo sin visitar, esta se sustituye con la menor distancia y se marca como visitado el nodo de **a**. Se toma como próximo nodo el menor valor en D y se repite el proceso.

Se importó el módulo heapq para así implementar las funciones .heappush() y .heappop(), para marcar los vértices. Además se creó un método getRoute para almacenar el camino para llegar al camino óptimo que encontraba el Dijkstra, en donde se tenía en cuenta el vértice anterior al vértice actual y la ruta se almacena en una lista.

Para encontrar el menor peso, implementamos la creación de la nueva variable v = (d*r)/2, donde **d** representa la distancia y **r** el riesgo de abuso. Se almacenaron los pesos resultantes de la creación de esta nueva variable y este fue el indicador para encontrar el camino óptimo con el algoritmo Dijkstra.

El algoritmo Dijkstra va de un nodo de origen hasta sus nodos adyacentes en donde evalúa cuál tiene menor peso (que equivale a la distancia/nivel de abuso), así iterativamente sumando los pesos para evaluar el recorrido por todos los nodos y encontrar el menor peso para llegar a la solución. Si el vértice es equivalente al destino el programa se detiene ya que llegó al objetivo . El algoritmo se ejemplifica en la Figura 3.

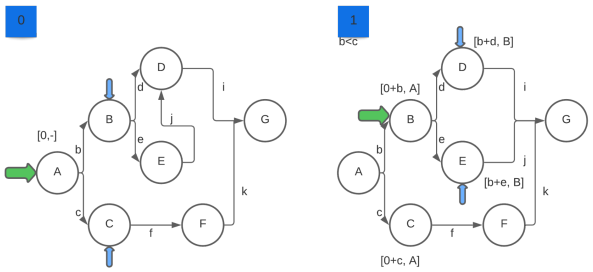


Figura 3: Cálculo de un camino que reduce tanto la distancia como el riesgo de acoso.

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

Los caminos fueron determinados por 3 variables que combinaban el riesgo y el acoso. La primera variable v=d*r le da la misma prioridad a ambas variables. La segunda variable v=d/100+r hace que la distancia se reduzca y sume el peso con este nuevo valor de la distancia y el mismo valor de riesgo de acoso. Además, la tercera variable v=d+10*r hace que se acelere el valor del riesgo de acoso por 100 y se lo suma a la distancia normal. El algoritmo se ejemplifica en la Figura 4.

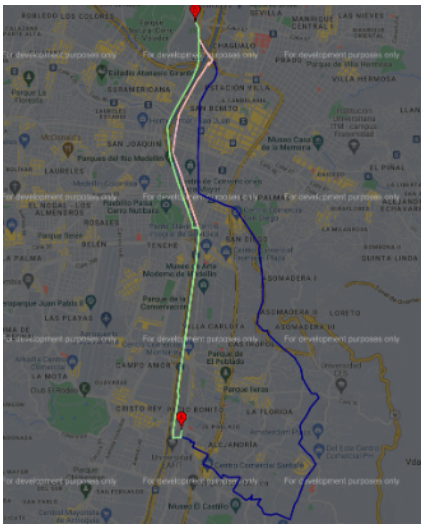


Figura 4: Mapa de la ciudad de Medellín donde se presentan tres caminos para peatones que reducen tanto el riesgo de acoso sexual como la distancia en metros entre la Universidad EAFIT y la Universidad Nacional.

4.3 Análisis de la complejidad del algoritmo

El algoritmo de Dijkstra implementado con una lista de adyacencia tiene una complejidad de O(E + V log V). El peor caso sería que tenga que recorrer todos los nodos que se deben añadir a la lista, la complejidad de añadir un elemento a una cola de prioridad es O(log V). El peor de los casos para una lista de adyacencia es que deba recorrer los n elementos.

Algoritmo	Complejidad temporal
Algoritmo de Dijkstra	O(E + V Log V)

Tabla 1: Complejidad temporal del algoritmo de Dijkstra, donde V representa el número de vértices del grafo y E el número de aristas.

Estructura de datos	Complejidad de la memoria
Lista de Adyacencia	O(V)

Tabla 2: Complejidad de memoria de la lista de adyacencia, donde V representa el número de vértices del grafo.

4.4 Criterios de diseño del algoritmo

El algoritmo de Dijkstra es un algoritmo muy común que se puede implementar de muchas maneras, debido a nuestra necesidad de combinar las variables de riesgo y abuso decidimos adaptar el que más útil y sencillo de implementar fuera. El algoritmo de Dijkstra es un algoritmo muy óptimo que encuentra el camino más corto y que minimiza demasiado el tiempo de búsqueda, por lo que era un algoritmo óptimo para la ocasión.

Además del Dijkstra, era necesario almacenar el recorrido, para después junto a la librería gmplot de python graficar los recorridos en google maps.

El algoritmo se diseñó de manera que fuera sencillo de programar, con herramientas ya conocidas que se desarrollaron a lo largo del curso y que pudimos encontrar en internet, para adaptar esto a nuestras necesidades. El código es funcional y fue realizado a nuestro alcance en base a los conocimientos que tenemos.

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	14394,3 m	0,33
Eafit	Unal	8080,76 m	0,55
Eafit	Unal	7889,8 m	0,61

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.

Cálculo de v	Tiempos medios de ejecución (s)
v = d*r	0,065 s
v = d/100 + r	0,06 s
v = d + 10*r	0,07 s

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

6. CONCLUSIONES

Realmente los caminos si son diferentes, aunque para todos los tiempos de ejecución son óptimos, podríamos compararlos y ver que para algunos cuando aumenta la seguridad la relación es inversamente proporcional con la distancia. El usuario podría elegir qué prefiere, si ir por un camino más seguro pero más largo o viceversa, o por el camino del medio que está entre los valores de los extremos. Medellín, como mencionamos en la introducción, es una ciudad insegura, y para las personas la seguridad se convirtió en una prioridad. En nuestra opinión, el camino de la variable v= d/100 + r es el camino que recomendamos, pues es una distancia no tan larga con un riesgo no tan alto. El código es funcional, con una idea muy importante que esperamos desarrollar ya que es una aplicación muy útil para las dinámicas sociales que estamos viviendo en los últimos tiempos.

6.1 Trabajos futuros

Es un trabajo muy interesante, y que perfectamente se puede aplicar a la vida real. En un futuro nos gustaría que

fuera más amigable con el usuario, que no se busque por coordenadas sino que funcione más como las dinámicas de google maps. Además, profundizar en la investigación de cuál sería la manera óptima de combinar el riesgo y la distancia para encontrar una variable en donde exista un balance entre estos.

AGRADECIMIENTOS

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. María Belén Fierro López, Pablo Joel López Jiménez, Libertad Machado López, Mariuxi Paola Cedeño Floril. *El acoso callejero, una forma de violencia contra la mujer*, Revista Metropolitana de Ciencias Aplicadas, 2020. 2-3: http://remca.umet.edu.ec/index.php/REMCA/article/view/239/281
2. Laura Solís Bastos. *Acoso sexual callejero, ¿no es para tanto o es para mucho? Percepciones sobre la violencia contra las mujeres en Costa Rica*, Repertorio Americano, 2018. 1-2: file:///C:/Users/Usuario/Downloads/11678-Texto%20del%20art%C3%ADculo-42290-1-10-20190403.pdf
3. Alcaldía de Medellín. *Se fortalecen las acciones de prevención de violencia y el acoso contra las mujeres en el centro de Medellín*, Alcaldía de Medellín, 2 de mayo de 2021: https://www.medellin.gov.co/es/sala-de-prensa/noticias/se-fortalecen-las-acciones-de-prevencion-de-la-violencia-y-el-acoso-contra-las-mujeres-en-el-centro-de-medellin/
4. Miguel López Mamani. *DFS vs BFS*, Encora, 25 de mayo 2020: https://www.encora.com/es/blog/dfs-vs-bfs#:~:text=Una%20b%C3%BAsqueda%20en%20profundidad%20(DFS,padre%20hacia%20el%20nodo%20hijo).
5. IAGraph. *Algoritmos de Búsqueda*, IAGraph, 2022: http://www.dma.fi.upm.es/personal/gregorio/grafos/web/iagraph/busqueda.html
6. Autor desconocido. *Algoritmos para la ruta más corta en un grafo*, desconocido, 2022: https://www.udb.edu.sv/udb_files/recursos_guias/informatica-ingenieria/programacion-iv/2019/ii/guia-10.pdf
7. EcuRed, *Algoritmo de Dijkstra*, EcuRed, 2022: https://www.ecured.cu/Algoritmo_de_Dijkstra
8. Algorithms and More. *Camino más corto: Algoritmo de Dijkstram*, Algorithms and More, 19 marzo 2022: https://jariasf.wordpress.com/2012/03/19/camino-mas-corto-algoritmo-de-dijkstra/
9. Instituto Tecnológico de Nuevo Laredo. *ALGORITMO A**, 2005: http://www.itnuevolaredo.edu.mx/takeyas/apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/A-Star/A-Star(2005-II-A).pdf
10. Ingrid Mendoza y Luisa Ruedas. *Algoritmo de Floyd-Warshall*, 2021: https://medium.com/algoritmo-floyd-warshall/algoritmo-de-floyd-warshall-e1fd1a900d8
11. Chacon, J. L. *Introducción a Pandas, la librería de Python para trabajar con datos*, Profile, 2022: https://profile.es/blog/pandas-python/
12. (s.f.), *Listas de Adyacencia*, hmong, 2022: https://hmong.es/wiki/Adjacency_list
13. DelftStack. *Lista enlazada en Python*, DelftStack, 2022: https://www.delftstack.com/es/howto/python/linked-list-in-python/
14. Unipython. (s.f.). *Grafos en Python*, Unipython, 2022: https://unipython.com/grafos-en-python/#:~:text=Un%20grafo%20es%20un%20conjunto,se%20llama%20grafo%20no%20dirigido
15. Autor desconocido. *Algoritmo de Dijkstra*, desconocido, 2022: https://nodo.ugto.mx/wp-content/uploads/2018/08/Dijkstra.pdf

16. Londoño, P. *Diccionario Python: todo lo que necesitas para crearlo y editarlo*,LECTURA DE 7 MIN,2022: https://blog.hubspot.es/website/diccionario-python

17. Autor desconocido. *Rutas más cortas de fuente única: algoritmo de Dijkstra*, Techie Delight, 2022: https://www.techiedelight.com/es/single-source-shortest-paths-dijkstras-algorithm/

18. Autor desconocido, *Shortest Path with Dijkstra's Algorithm*, bradfieldcs, 2022: https://bradfieldcs.com/algos/graphs/dijkstras-algorithm/

19. Autor desconocido. Plotting Google Maps using gmplot package in Python, JavaTPoint, 2022: https://www.javatpoint.com/plotting-google-map-using-gmplot-package-in-python