ALGORITMO PARA LA REDUCCIÓN DE DISTANCIA Y ACOSO SEXUAL EN MEDELLIN

Miguel Ángel Martínez García
Universidad Eafit
Colombia
mamartin11@eafit.edu.co

Camilo Enrique Soto Reyes
Universidad Eafit
Colombia
cesotor@eafit.edu.co

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

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

RESUMEN

La inseguridad en la ciudad de Medellín es un problema que por décadas ha afectado a las personas que circulan por sus calles. Pese al esfuerzo de la alcaldía por disminuir sus cifras, en algunas zonas de la ciudad, el índice de inseguridad ha aumentado hasta un 24%. Este problema ha permanecido en la ciudad por un gran periodo de tiempo, haciendo que el desarrollo de la ciudad aumente a una menor escala, permitiendo así que los actos ilegales en las calles aumenten.

Es de gran importancia que se trate este problema debido a que este no permite a los ciudadanos convivir en paz y tranquilidad, por el contrario, la inseguridad en Medellín solo aumenta el miedo a circular libremente por las calles de la ciudad. La inseguridad en la calle se ve reflejada en los diferentes actos ilegales que suceden en la ciudad; robos, acoso sexual, secuestros, extorciones e incluso homicidios, son algunos de los factores los cuales componen la inseguridad en la ciudad.

Es por esto, que se planteó la idea de realizar un algoritmo basado en datos obtenidos, que permita arrojar la ruta más corta para que una persona se movilice de un punto A a uno B y así evitar los problemas que trae consigo la inseguridad como el acoso callejero.

Palabras clave

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

1. INTRODUCCIÓN

Explique la motivación, en el mundo real, que conduce al Inicialmente, es de suma importancia realizar un algoritmo que permita identificar la ruta más corta entre dos puntos de la ciudad, debido a que muchos de los ciudadanos toman rutas inadecuadas o rutas más largas exponiéndose así ante peligros que la ciudad tiene. Por otro lado, es importante realizar esta solución debido a que la tecnología actualmente cuenta con un nivel de identificación y solución de problemas en tiempos muy cortos y con resultados realmente eficientes y eficaces.

Debido a esto, es necesario la implementación de este algoritmo para reducir el índice de inseguridad de la ciudad de Medellín. Además de esto, el algoritmo permitirá a las personas estar menos tiempo expuestas a ser víctimas de acoso sexual, para ofrecer así una mejor seguridad y una mayor tranquilidad a todos los ciudadanos.

1.1. Problema

EL problema del acoso sexual callejero se da en gran medida a la mala elección de las personas a la hora de tomar una ruta para ir de un punto a otro, generando así una mayor posibilidad de que se genere un acoso sexual principalmente en mujeres.

Es por esto por lo que es importante dar solución al problema por medio de un algoritmo, para así reducir el número de acosos sexuales callejeros, además de proporcionar rutas más cortas y seguras para los usuarios.

1.2 Solución

El algoritmo elegido para dar solución al problema es el algoritmo Dijkstra. Este algoritmo permite ingresar un nodo inicial el cual será el lugar de origen del recorrido hasta un nodo final que sea definido. Por otro lado, este algoritmo funciona perfectamente en grafos ponderados, siendo esta la estructura en la que se definirá el mapa de la ciudad de Medellín, y siendo la variable v, el peso de la arista que conecta los nodos la cual combina el acoso callejero con la distancia. Además de esto, la implementación del algoritmo Dijkstra es muy intuitiva, permitiendo así comprender y analizar el código de una mejor manera.

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 Algoritmo optimización de rutas para servicio courier.

El artículo da solución al problema de las malas rutas de entrega de una empresa que ofrece servicio Courier. Para esto utilizaron el algoritmo Dijsktra para obtener las rutas más cortas. Se obtuvieron resultados muy positivos puesto que lograron reducir el porcentaje de entregas impuntuales en un 50% y generando una mayor satisfacción en el cliente

2.2 Algoritmo para optimización de rutas y caminos cortos.

Se planteó un planificador de rutas para optimizar la forma en la que se recogen los desechos sólidos a lo largo de una ruta entre dos puntos. Para esto se utilizó el algoritmo de Bellman Ford. Se obtuvieron tiempos cortos en los cuales se realiza la recolección de los deshechos, así como también optimizar rutas en diferentes situaciones que se presenten en la vida real.

2.3 Algoritmo de abejas para adaptación.

En este artículo se dio solución a la problemática de ahorro de tiempo para llevar los pedidos de una mejor forma a los clientes. Para esto se utilizó el algoritmo de colonia de abejas artificiales. Se obtuvieron resultados positivos en cuanto a la adaptación del algoritmo con las condiciones del entorno de manera eficiente y con un error mínimo.

2.4 4 Ruta más corta entre dos puntos geográficos.

Este proyecto busca encontrar la ruta más corta entre dos puntos geográficos por medio de la comparación de tres algoritmos: Dijsktra, Bellman Ford y Floyd Warshall. Se obtuvo que el algoritmo Dijsktra arrojó un menor tiempo y una mayor cantidad de rutas.

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 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].

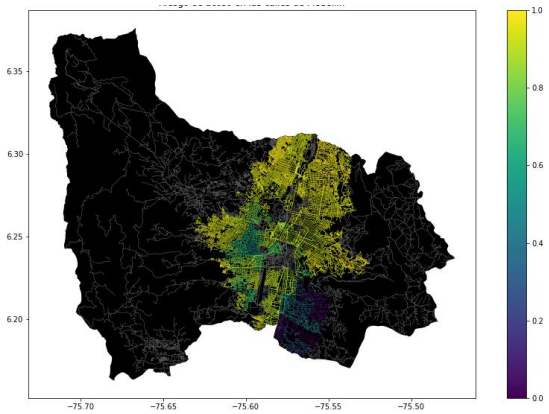


Figura 1. Riesgo de acoso sexual calculado como una combinación lineal de la fracción de hogares que se sienten inseguros y la fracción de hogares con ingresos inferiores a un salario mínimo, obtenidas de la Encuesta de Calidad de Vida de Medellín, de 2017.

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. *(En este semestre, ejemplos de dichos algoritmos son DFS, BFS, Dijkstra, A*, Bellman, Floyd, entre otros).*

3.2.1 Algoritmo Dijjsktra

Este algoritmo es un algoritmo iterativo que permite encontrar el camino más eficiente desde un nodo inicial hacia

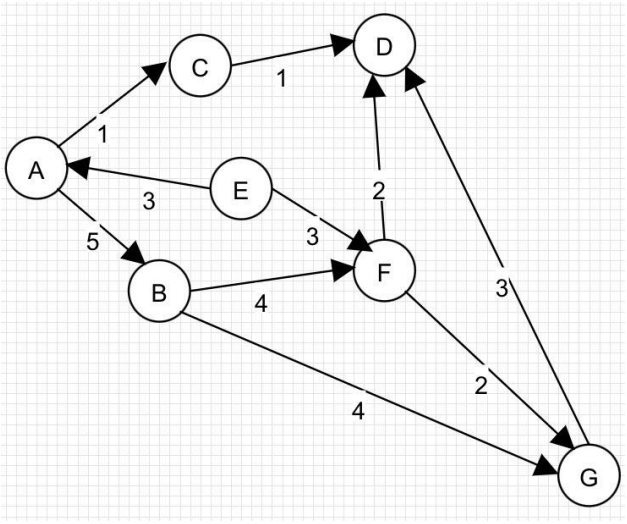
[1] https://www.openstreetmap.org/

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

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

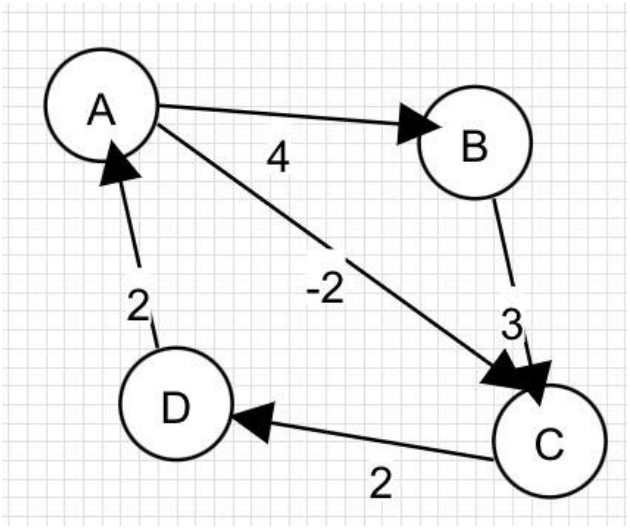
los demás nodos, siendo este un algoritmo de búsqueda. No permite valores negativos en los arcos.

Este algoritmo funciona bajo el principio de optimalidad el cual menciona que “si el camino más corto entre los vértices u y v pasa por el vértice w, entonces la parte del camino que va de w a v debe ser el camino más corto entre todos los caminos que van de w a v”. Este algoritmo toma un grafo dado el cual tiene un valor definido en sus arcos, por medio de este valor, el algoritmo busca recorrer el camino más definiendo así un coste mínimo de un vértice a otro entre todos los posibles caminos que se puedan generar entre aquellos dos vértices.



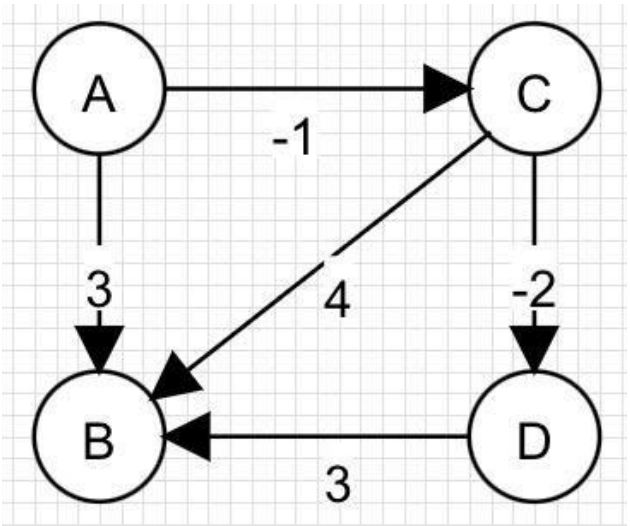
3.2.2 Algoritmo de Bellman Ford

Este algoritmo busca el camino más corto desde un nodo de origen de grafo ponderado hacia los demás nodos, este permite tener valores negativos en los arcos, por lo que se presenta una solución más general. A pesar de que implementar una solución con este algoritmo puede ser más lenta que una obtenida por medio del algoritmo Dijsktra, el algoritmo Bellman permite obtener como resultado un ciclo de ruta negativo, iterando hasta que la distancia entre los vértices analizados sea mínima, recorriendo así, todos los bordes del grafo dado en un orden aleatorio.



3.2.3 Algoritmo de Floyd-Warshall

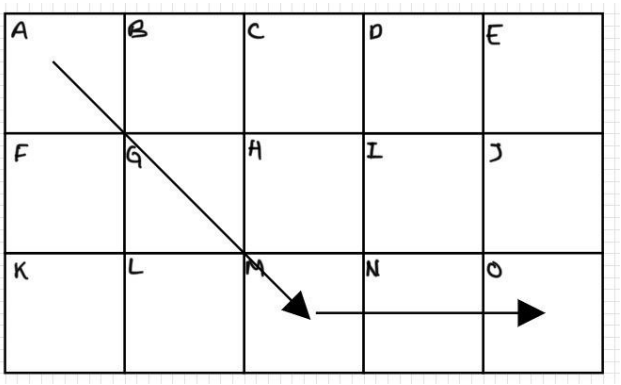
Algoritmo utilizado para hallar el camino más corto entre cualquier par de nodos, lo cual significa que no necesita un nodo principal de origen, tal como lo es necesario para el Dijsktra. En comparación con el algoritmo de Bellman, este algoritmo también acepta tanto valores positivos como negativos de pesos de una ruta, sin embargo, este algoritmo no implementa los ciclos negativos. Este algoritmo revisa las longitudes y costos de todos los posibles caminos entre todos los pares de vértices de manera iterativa.



3.2.4 Algoritmo A*

Este algoritmo busca el camino más corto entre nodos por medio de una función que utiliza el valor heurístico del nodo a evaluar, desde el nodo inicial hasta el final por medio del costo de ruta. Este algoritmo utiliza una heurística óptima, puesto que esta no sobrestima la distancia real entre el nodo inicial y el final. Sin embargo, una heurística puede ayudar a

simplificar la solución de un problema en caso de que este sí tenga solución, puesto que no garantiza que este sea solucionado.



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 Estructura de datos

Para representar el mapa de la ciudad de Medellín, se utilizó una lista de adyacencia utilizando un diccionario. En esta estructura, las llaves serán las calles de la ciudad (no los segmentos de las calles) y los valores estarán compuestos por dos listas. La primera lista contendrá las calles que conectan con la llave (incluyendo sus segmentos) y la segunda lista contendrá la variable v que relaciona la llave con cada posición de la primera lista. La estructura de los datos se presenta en la figura 2.

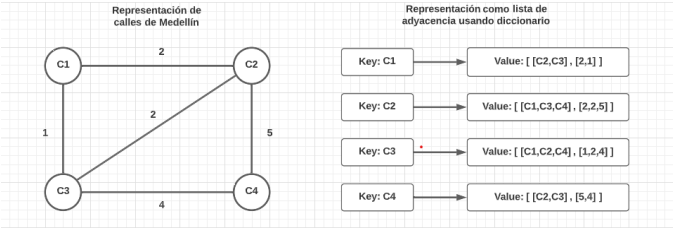


Figura 2.

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 comprender el algoritmo Dijkstra, se explica el proceso que este realiza a continuación:

a. Se asigna un nodo inicial con una distancia (peso) de 0 y a los demás nodos se les asigna una distancia de ‘infinito’.
b. Se asigna un predecesor nulo para todos los nodos.
c. Se establece el nodo inicial como el actual y se crea un array de nodos no visitados que contendrá a todos los nodos.
d. Mientras que el array tenga elementos:
e. Se toma el nodo actual u, con sus vecinos no visitados v y pesos de arista w:
   A. Si la distancia del nodo actual u, sumada con el peso w es menor a la distancia del nodo v, se actualiza la distancia del nodo v y se guarda a u como el predecesor de v.
f. Una vez se revisan todos los vecinos de u, este se marca como visitado y se elimina del array de nodos no visitados.
g. Se selecciona el nodo no visitado con menor distancia y se toma como el nuevo nodo actual.
h. Se regresa al paso d.

El algoritmo se ejemplifica en las figuras 3,4,5 y 6.

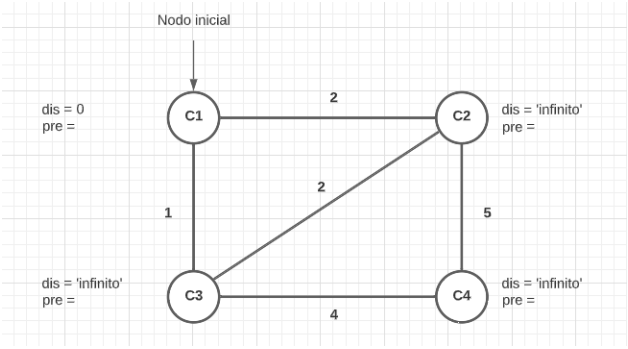


Figura 3. Estado inicial del algoritmo, pasos a, b y c.

[4] https://github.com/mamartin11/proyecto

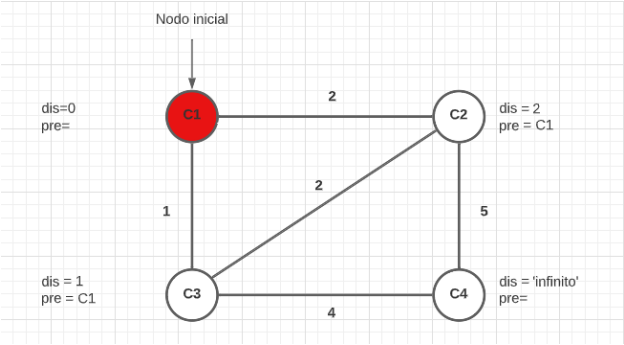


Figura 4. Estado del algoritmo después de realizar los pasos a, b, c, d, e, f y g por primera vez.

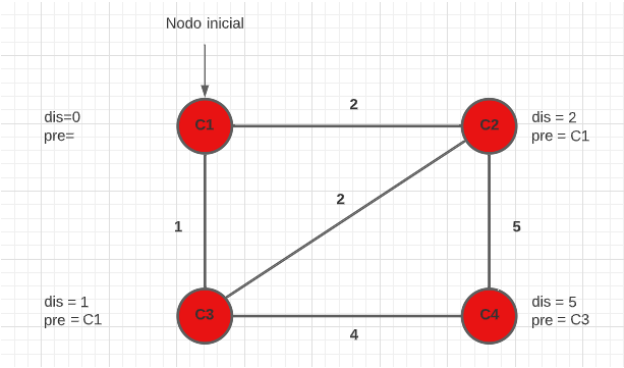


Figura 5. Estado final del algoritmo después de realizar todas las iteraciones del paso d.

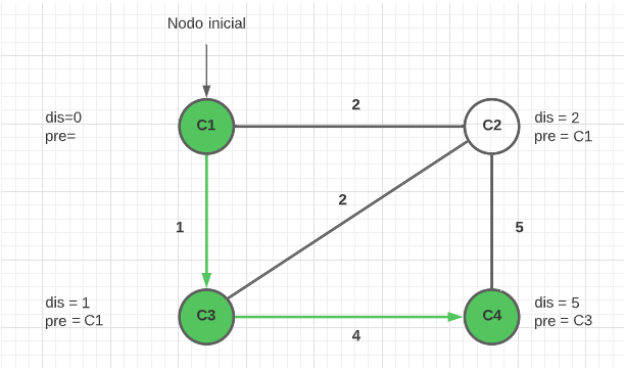


Figura 6. Se presenta el camino más corto de C1 a C4.

REFERENCIAS

1. Aguilar, C. Algoritmos de búsqueda para la mejor ruta basado en grafos. Retrived August 22, 2022, from Dspace repository:

   http://repositoriodigital.tuxtla.tecnm.mx/xmlui/handle/123456789/84

2. Grigelmo, G. Applet de algoritmo de Bellman-Ford. Retrived August 20, 20022, from BURJC digital repository:

   https://burjcdigital.urjc.es/handle/10115/3365

3. Milian, J.L. Sistema web basado en algoritmo de ruta más corta para optimización de rutas en la empresa de servicios logísticos de courier Seminario Martínez Servicios Generales S.A.C. Retrived August 21,2022, from USAT repository:

   https://tesis.usat.edu.pe/handle/20.500.12423/2237

4. Pérez, R.C. Algoritmo adaptativo para la selección de la ruta más corta. Retrived August 20, 2022, from Dspace repository:

   http://repositoriodigital.tuxtla.tecnm.mx/xmlui/handle/123456789/2046

5. Penagos, N.L. Análisis e implementación de un algoritmo incrustado en un administrador de base de datos relacional, para encontrar la ruta más corta entre dos puntos geográficos. Retrived August 22, 2022, from UVG digital repository:

   https://repositorio.uvg.edu.gt/handle/123456789/1546

6. Reyes, J.O. DESARROLLO DE UN PLANIFICADOR DE RUTAS PARA RECOJO DE DESECHOS SÓLIDOS UTILIZANDO ALGORITMO DE BELLMAN FORD. Retrived August 21,2022, from USS repository:

   https://repositorio.uss.edu.pe/bitstream/handle/20.500.12802/5779/Reyes%20Esqu%C3%A9n%20Jeremy%20Octavio.pdf?sequence=1

7. Sánches, G., Lozano, V.M. Algoritmo Dijkstra. Un tutorial interactivo. Retrived August 22, 2022.

   http://bioinfo.uib.es/~joemiro/aenui/procJenui/ProcWeb/actas2001/saalg223.pdf

Link para ver el video y la presentación:

https://github.com/mamartin11/proyecto