FINDING THE SHORTEST PATH PREVENTING SEXUAL HARASSMENT THROUGH ALGORITHMS

Juan Felipe Restrepo Buitrago
Universidad Eafit
Colombia
jfrestrepb@eafit.edu.co

Sara Valentina Cortes Manrique
Universidad Eafit
Colombia
svcortesm@eafit.edu.co

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

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

ABSTRACT

Sexual harassment is a daily concern for Medellin's women. Thanks to Medellin's mayoralty which surveyed 1000 women, we know that 85% of them have suffered sexual harassment [19]. We have to take control of this situation, due to the insecure feeling present in most women, so they could live more calmly and comfortably going anywhere. With this project, we hope to create a solution to this problem using an algorithm to find a safer path that takes the least time possible to get from one location to another, this algorithm consists on finding the path with the lowest numeric amount between two points in a graph. Trying to avoid sexual harassment on the streets is impossible without considering crime, which helps men and women feel less comfortable out in the city.

Keywords

The shortest route, street sexual harassment, identification of safe routes, crime prevention

1. INTRODUCTION

As we said lately, people don't feel safe in the city and are worried, especially women. If the city were safer, surely our parents hadn't have taught us to be extremely cautious on the streets, as the majority probably are. According to the given data by The Legal Medicine National Institute, in the first trimester of 2022, we had 6.336 violent homicides, 848 more than the first quarter of 2021. Medellín is one of the most affected cities in the country by this matter, with around 232 homicides [11]. Although the number of cases in Medellin has been reduced, people are still worried chiefly about going out at night with 35% of safety according to what Medellín's people think about crime in the last 3 years [8]. We want people to feel safer and more comfortable while going on the street.

1.1. The problem

The problem we're trying to solve is to find three paths to lead people to their destinies. One of them will be the shortest without considering the danger of the path it is going through, the other one will be the safest without considering the distance, and the last one will consider both distance and safety proportionally. These three paths are important because of the situation the person using the algorithm is living through. In the first place, if they need to hurry and get as quickly as possible to a place, they will probably avoid the safety parameter, on the other hand, someone could probably not need time, but safety or another person could need both parameters in their trip.

1.2 Solution

Our solution to this problem is to implement an algorithm that finds the shortest and safest path to go from one place to another in Medellin. We chose Dijkstra's algorithm to find the shortest path to a destination, which goes from one specific starting node to all other nodes in a weighted and directed graph. Also, we chose it because of its easy implementation and its execution time, which is not very high.

To determine which path is the most convenient, we must consider distance and harassment risk. For this algorithm we will find three cases: a route with the shortest path, a route with the safest path, and a route that considers both the safest and shortest path at the same time, to get a balance between these two variables, so we can guarantee the safety of women through the shortest route possible.

1.3 Structure of the article

Next, in Section 2, we present work related to the problem. Then, in Section 3, we present the datasets and methods used in this research. In Section 4, we present the algorithm design. Then, in Section 5, we present the results. Finally, in Section 6, we discuss the results and propose some directions for future work.

2. RELATED WORK

Below, we explain four works related to finding ways to prevent street sexual harassment and crime in general.

2.1 Safetipin: A Free Map-Based Application Helps Users to Which Areas They Would Like to Pass Through and Which Ones to Avoid.

Safetipin is a mobile application that allows people to check whether a location is safe or not. It finds the best paths to take people to a location that avoids unsafety places where dangers such as crime and street sexual harassment could occur. Although the safety of somewhere is mainly calculated by Safetipin's team, users can give their opinions of a place, improving the score's accuracy of those places. Besides, people can tell the app which places they want to avoid. Safetipin was created by Kalpana Viswanath to treat the safety problem in Delhi [16].

Talking in algorithm terms, they are based on GIS to collect their information and their application runs machine learning [17].

2.2 The Safe Route: Multi-Options Route Finder for Cyclists.

The Safe Route is an application developed especially for bike drivers. This app provides the user multiple options of paths he could go through from the fastest to the largest and safest in terms of traffic and accidents. The Safe route was created by the company Futurice. The problem this app is considering is road insecurity for cyclists in Sweden. They hope to encourage people to ride a bike, giving them safe ways and helping the environment [12].

The parameters this app considers for providing navigation based on safety are traffic jams, road work, crossings, poor surfaces, weather, and accident statistics [10] Data about the kind of algorithm this app uses are not given.

2.3 TomTom: Algorithms Prioritizing Safety Over Speed

TomTom is an application, which finds routes to take people from one location to another. Lately, this app is taking into account the safety of some roads above others. This new function was implemented because of the danger drivers were exposed to, one study was made in Finland that the fastest route to Koli National Park, was the most dangerous mainly in winter, because of the snow. So now this app takes parameters such as weather, quality of the road, etc. for finding a path [1].

TomTom's engine is based on the A* algorithm [17].

2.4 Path Community: Red Flags on the Streets

Path is an application that suggests routes in which the user is not likely to be involved in a harassment, assault, or attack situation. It was created by Harry Mead. This is a user's opinion-based application, where a user can highlight a dangerous area as a red flag, which helps the app find the safest route from one point to another by avoiding these red flags [13].

3. MATERIALS AND METHODS

In this section, we explain how the data were collected and processed, and then different alternative path algorithms that reduce both the distance and the risk of sexual street harassment.

3.1 Data collection and processing

The map of Medellín was obtained from *Open Street Maps* (OSM)[1] and downloaded using the Python API[2] OSMnx. The map includes (1) the length of each segment, in meters; (2) the indication of whether the segment is one-way or not, and (3) the known binary representations of the geometries obtained from the metadata provided by OSM.

For this project, a linear combination (LC) was calculated that captures the maximum variance between (i) the fraction of households that feel insecure and (ii) the fraction of households with incomes below one minimum wage. These data were obtained from the 2017 Medellín quality of life survey. The CL was normalized, using the maximum and minimum, to obtain values between 0 and 1. The CL was obtained using principal components analysis. The risk of harassment is defined as one minus the normalized CL. Figure 1 presents the calculated risk of bullying. The map is available on GitHub[3] .

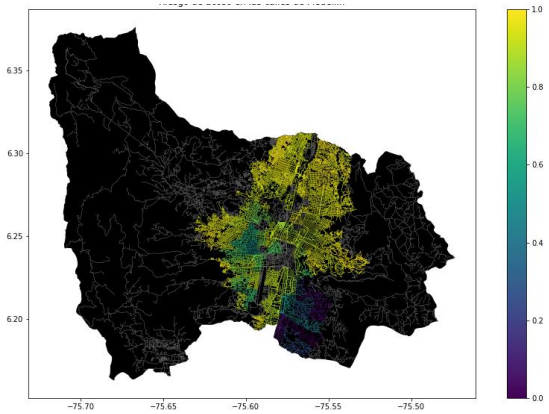


Figure 1. Risk of sexual harassment calculated as a linear combination of the fraction of households that feel unsafe and the fraction of households with income below one minimum wage, obtained from the 2017 Medellín Quality of Life Survey.

3.2 Algorithmic alternatives that reduce the risk of sexual street harassment and distance

In the following, we present different algorithms used for a path that reduces both street sexual harassment and distance.

3.2.1 Breadth-First Search

Is an important graph search algorithm that is useful for analyzing and solving graph problems as our problem, finding the shortest path. [3]

BFS parses each node and edge of a graph using a queue and something that tells us whether an edge has already been visited or not. As his name says, this algorithm analyzes all

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

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

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

[4] https://github.com/JuanFelipeRestrepoBuitrago/ST0245

the adjacent nodes of a specific vertex, which in a graph could be seen as the first level, then it continues to the next level until it reaches the final node. When a final node has been reached by the BFS, it will return a data structure that contains the shortest path. This algorithm claims that the first time a node is discovered during the traversal, the distance from the source would give us the shortest path to that node [9].

The complexity for BFS algorithm is O(V+E), where V is the number of vertices and E the number of edges [3].

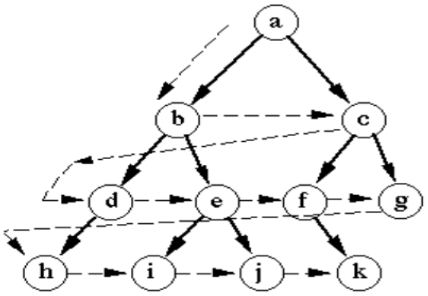


Figure 2 Breadth First Search Algorithm [14].

3.2.2 Depth First Search

Abbreviated as DFS, this algorithm is very similar to the BFS algorithm. DFS is about fully parsing an adjacent node before parsing another one, meaning that it goes through all the nodes of an adjacent node, and then trackbacks until it finds an unexplored path, the next adjacent node of the starting node, and explores it. While BFS is guaranteed to return an optimal answer, DFS is not.

The complexity for DFS algorithm is O(V+E), where V is the number of vertices and E the number of edges [4].

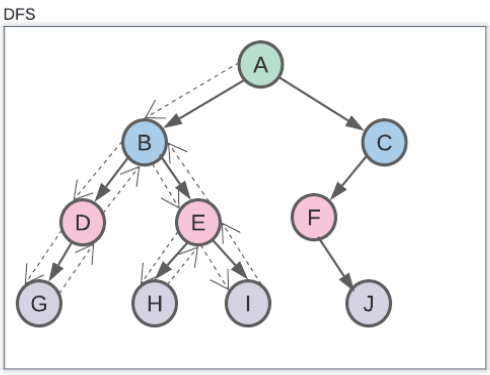


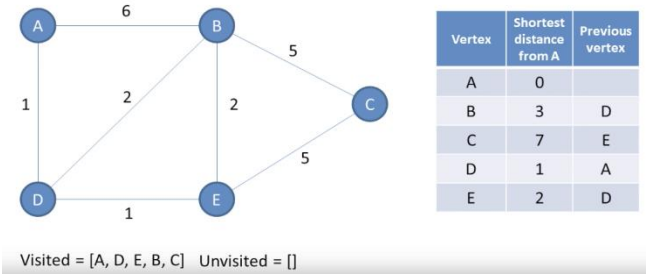
Figure 3 Depth First Search [7].

3.2.3 Dijkstra's Algorithm

It is an algorithm to find the shortest path from a starting node to a target node in a weighted graph. The graph can either be directed or undirected. One stipulation to using the algorithm is that the graph needs to have a nonnegative weight on every edge.

Dijkstra Algorithm starts from the starting node and parses the distances to each connected node of the graph, in other words, the weight between nodes. It constantly builds a set of nodes that have a minimum distance from the source repeating this process until all nodes have been visited and the destination node has been reached. It also returns a data structure that contains the shortest path from one node or vertex to another [5].

The complexity for Dijkstra algorithm implementing a priority queue is O((V+E) log V), where V is the number of vertices and E the number of edges [15].



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Figure 4 Dijkstra's Algorithm [6].

3.2.4 Bellman-Ford Algorithm

The Bellman-Ford Algorithm is a graph search algorithm that finds the shortest path between a given source vertex and all other vertices in the graph. This algorithm can be used on both weighted and unweighted graphs.

Like Dijkstra's shortest path algorithm, the Bellman-Ford is guaranteed to find the shortest path in a graph. Though it is slower than Dijkstra's algorithm, Bellman-Ford is capable of handling graphs that contain negative edge weights, so it is more versatile.

This algorithm first set all the distances to go to the source node to this the final node. Unlike Dijkstra, Bellman-Ford Algorithm doesn’t need to verify whether a node has been visited or not. While setting the distances of each node, it also set its predecessor. When the algorithm eventually went through all nodes of the graph, the shortest path will be in the predecessor dictionary.

The complexity for Bellman-Ford algorithm is O (V.E), where V is the number of vertices and E the number of edges [2].

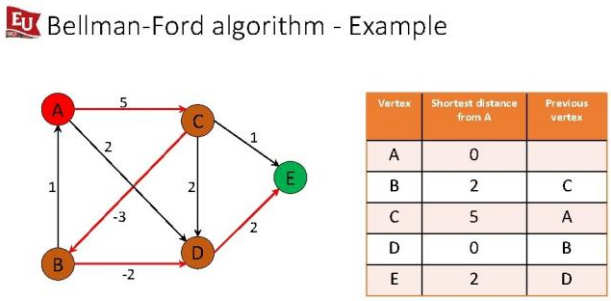


Figure 5. Bellman-Ford Algorithm [18].

4. ALGORITHM DESIGN AND IMPLEMENTATION

In the following, we explain the data structures and algorithms used in this work. The implementations of the data structures and algorithms are available on Github[4] .

4.1 Data Structures

The data was given via CSV file. We used Pandas DataFrame to extract the data from the file. As a data structure, we were using a graph and, in our implementation, the representation of the graph is an Adjacency List and we used python dictionaries to implement it. Each key of the dictionary or the graph is a coordinate extracted from the Pandas DataFrame and the values for these keys are the adjacent coordinates (adjacent nodes) which are also dictionaries with coordinates as keys and distance and harassment risk as values within a tuple. *The data structure is presented in Figure 7.*

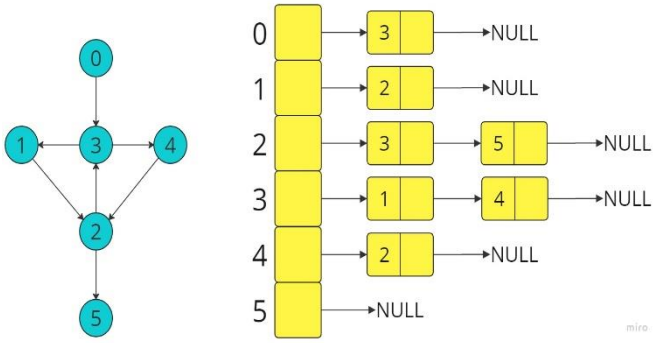


Figure 6 Graph Represented as Adjacency List.

4.2 Algorithms

In this paper, we propose an algorithm for a path that minimizes both the distance and the risk of street sexual harassment.

4.2.1 Dijkstra Algorithm

Since we must find the shortest path on a weighted and directed graph, which considers the distance and harassment risk, we chose Dijkstra Algorithm to implement our solution.

We created a dictionary that will contain all distances from a source node to all other nodes. In the beginning, the distance to all other nodes has a value of positive infinity, except the distance from the source node, which has a value of 0.

To find the shortest path we created a priority queue that contains the nodes with the shortest distances, which will always return the node with the shortest distance. Until our priority queue is empty, we will pop the node with the minor distance, then we check if the adjacent node to our current node is already visited, if it's true, we find the distance to that node from the current, then, if that distance is minor than previous values of distances, we set that distance as the new value and push that node into our priority queue with the nodes that have the shortest distances.

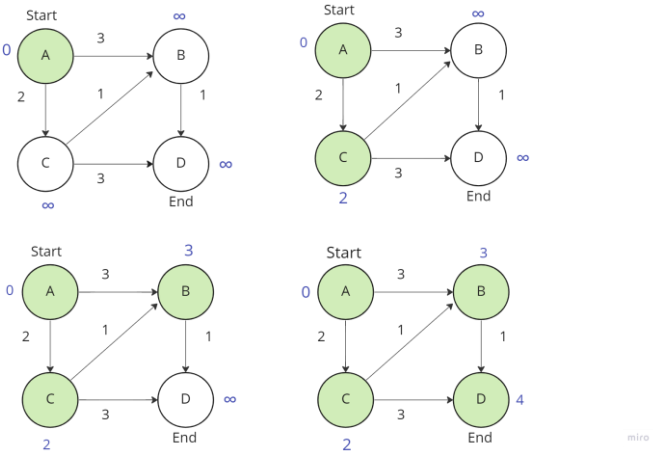


Figure 7 Dijkstra Algorithm

4.2.2 Calculation of two other paths to reduce both the distance and the risk of sexual street harassment

Each one of the three paths that were made by our algorithm, has a different related variable. For the shortest path, our variable was the distances between streets, the harassment risk for this first path didn't matter. For the safest path, our main variable was the harassment risk, but we found out that if we raised the value of the distance to the power with the value of the risk, we would get a lower harassment risk than only having in mind the risk, so our second variable was this operation mentioned before. For the third path, which was a combination of the last two paths, we took an average made

[4] https://github.com/JuanFelipeRestrepoBuitrago/ST0245

with the distance and the second used variable of each street as a variable. Once we had clear which were the three variables, we applied our implementation of Dijkstra to each one of them. The algorithm is exemplified in Figure 4.

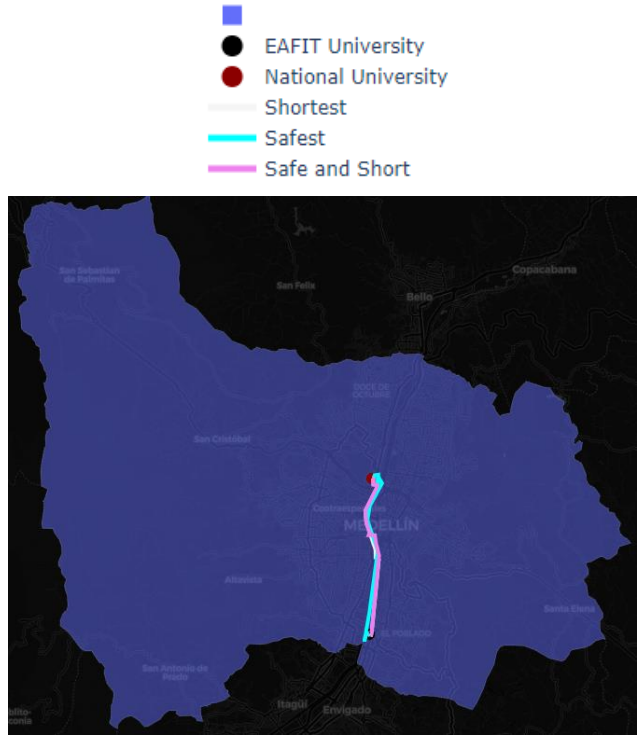


Figure 4: Map of the city of Medellín showing three pedestrian paths that reduce both the risk of sexual harassment and the distance in meters between the EAFIT University and the National University.

4.3 Algorithm complexity analysis

Our Dijkstra, which contains a standard heap and an adjacency list implementation, has a complexity of O ((E+V) log V), with V being the vertex of the graph and E the edges, in other words, V are each point of the graph and V are the streets between points. The worst case occurs when the while loop must pass through all nodes of the graph and in the meanwhile adding each of them to the priority queue. For adding and removing an element to the priority queue, the complexity is O (V log V) and in the worst case each edge will be checked twice, giving a complexity of O (E log V). If we merge the last 2 given complexities, we will have the complexity O ((E+V) log V). [15]

Algorithm	**Time complexity**
Dijkstra	O((V+E) log V)

Table 1: Time complexity of the Dijkstra algorithm, with V being the total number of nodes or points of the graph and E being the total number of connections between point, the streets in fact

Data Structure	**Complexity of memory**
Adjacency List	O(V)

Table 2: Memory complexity of the adjacency list, with V being the number of nodes of the graph or the points in the street.

4.4 Algorithm design criteria

Efficiency and easiness were our main matters while doing this project.

We decided to implement an easy menu to avoid almost totally the user's mistakes. With this Menu the user can select some locations and which paths they want to see in an easy way to do it.

We also chose to use an adjacency list made with python dictionaries, because of the easiness to access all the data and because is better the complexity of an adjacency list than an adjacency matrix.

For the first time we implemented the standard Dijkstra algorithm, but our execution times were too high. But with the priority queue execution, those times reduced very much. Now is very fast this process, it doesn't even take 1 second to return the paths.

For drawing our paths, we implemented a library called Plotly and Geopandas. First, we needed geopandas to get our GeoDataFrame, then Plotly just created some figures, which were painted by this library. Although this library returned the expected result, it was difficult to implement because it was a complex library, had a lot of functions and things that we had to investigate to understand.

5. RESULTS

In this section, we present some quantitative results on the three pathways, the shortest, the safest and a combination of both.

5.1 Results of the paths that reduces both distance and risk of sexual street harassment

With the following section, we present the distance and risk results of the three paths for 3 different locations. First is from EAFIT University to National University, the second one is from University of Antioquia to University of Medellín and the third one is from Bolivarian Pontifical University to Colombian Polytechnic Jaime Isaza Cadavid.

5.1.1 The Shortest Path Results

Next, we present the results obtained from three paths that reduce distance in Table 3.

Origin	Destination	Distance	Risk
Eafit University	National University	7686.62	0.71
University of Antioquia	University of Medellín	7615.75	0.83
Bolivarian Pontifical University	Polytechnic Jaime Isaza Cadavid	4610.0	0.83

Table 3: Distance in meters and risk of sexual street harassment (between 0 and 1) for the shortest path.

5.1.2 The Safest Path Results

Next, we present the results obtained from three paths that reduce harassment risk in Table 4.

Origin	Destination	Distance	Risk
Eafit University	National University	11027.69	0.47
University of Antioquia	University of Medellín	8636.11	0.76
Bolivarian Pontifical University	Polytechnic Jaime Isaza Cadavid	8583.72	0.57

Table 4: Distance in meters and risk of sexual street harassment (between 0 and 1) for the safest path.

5.1.3 The Safe and Short Path Results

Next, we present the results obtained from three paths that reduce both distance and harassment risk, a combination of these two variables, in fact, in Table 5.

Origin	Destination	Distance	Risk
Eafit University	National University	7762.26	0.72
University of Antioquia	University of Medellín	7937.52	0.77
Bolivarian Pontifical University	Polytechnic Jaime Isaza Cadavid	4614.33	0.82

Table 5: Distance in meters and risk of sexual street harassment (between 0 and 1) for the safe and short path.

5.2 Algorithm execution times

In Table 6, we explain the ratio of the average execution times of the queries presented in the last three tables.

Calculation	Average run times (s)
EAFIT University to National University	0.16
University of Antioquia to University of Medellín	0.24
Bolivarian Pontifical University to Polytechnic Jaime Isaza Cadavid	0.15

Table 6: Execution times of the Dijkstra algorithm for each of the queries presented in the last three tables.

6. CONCLUSIONS

First, this project could be improved with a graphic interface, where the user would do everything better and it will look better. Although our routes reduce the harassment risk a bit, it is still too high that almost all paths are rated up to 0.5, which is worrying mostly for women who reside in Medellín city and for the tourists, who are not so safe out there or perhaps do not want to come because of safety.

Lastly, we are very proud of the results we got because each of the paths we generated were appropriate to what we were expecting. Also, we feel comfortable with the execution times of each one of them.

6.1 Future work

We enjoyed working with something related to our city, it felt like we would really be helping with something in our city, we were doing something great. We think the best way to continue working with this project is to inset this algorithm into the applications world, where it would be of benefit to all users that the app would have. Furthermore, we would like to implement a location in real time function to let the user know how they are doing through that path and real time changes during the trip, because anything could happen in the middle of it.

ACKNOWLEDGEMENTS
The authors thank Professor Juan Carlos Duque, Universidad EAFIT, for providing the data from the 2017 Medellín Quality of Life Survey, processed in a Shapefile.

REFERENCES

1. Beedham, M., 2022. How routing algorithms prioritize safety over speed in rural Finland. [Blog] *TomTom*, Available at: <https://www.tomtom.com/blog/navigation/map-data-to-suggest-the-safest-route/> [Accessed 6 October 2022].
2. Brilliant.org. 2022. *Bellman-Ford Algorithm | Brilliant Math & Science Wiki*. [online] Available at: <https://brilliant.org/wiki/bellman-ford-algorithm/> [Accessed 6 October 2022].
3. Brilliant.org. 2022. *Breadth-First Search (BFS) | Brilliant Math & Science Wiki*. [online] Available at: <https://brilliant.org/wiki/breadth-first-search-bfs/> [Accessed 6 October 2022].
4. Brilliant.org. 2022. *Depth-First Search (DFS) | Brilliant Math & Science Wiki*. [online] Available at: <https://brilliant.org/wiki/depth-first-search-dfs/> [Accessed 6 October 2022].
5. Brilliant.org. 2022. *Dijkstra's Shortest Path Algorithm | Brilliant Math & Science Wiki*. [online] Available at: <https://brilliant.org/wiki/dijkstras-short-path-finder/> [Accessed 6 October 2022].
6. Computer Science, 2022. *Graph Data Structure 4. Dijkstra Shortest Path Algorithm*. [image] Available at: <https://youtu.be/pVfj6mxhdMw> [Accessed 6 October 2022].
7. Cortes, S., 2022. *DFS Representation example*. [image] Available at: <https://lucid.app/lucidspark/bcf35ea3-ce3d-42d2-b9d0-7b184688fb31/edit?invitationId=inv_7b61f247-0e94-4ff5-a133-cac609c1ad3f#> [Accessed 6 October 2022].
8. Es.numbeo.com. 2022. *Criminalidad en Medellín*. [online] Available at: <https://es.numbeo.com/criminalidad/ciudad/Medell%C3%ADn> [Accessed 6 October 2022].
9. FreeCodeCamp.org. 2018. Finding Shortest Paths using Breadth First Search. [online] Available at: <https://www.freecodecamp.org/news/exploring-the-applications-and-limits-of-breadth-first-search-to-the-shortest-paths-in-a-weighted-1e7b28b3307/#:~:text=And%20so%2C%20the%20only%20possible,source%20to%20the%20destination%20vertex> [Accessed 7 October 2022].
10. Futurice.com. 2022. *The Safe Route*. [online] Available at: <https://futurice.com/saferoute> [Accessed 6 October 2022].
11. Infobae. 2022. *Se registraron 6336 muertes violentas en Colombia en el primer trimestre de 2022*. [online] Available at: <https://www.infobae.com/america/colombia/2022/05/05/se-registraron-6336-muertes-violentas-en-colombia-en-el-primer-trimestre-de-2022/> [Accessed 6 October 2022].
12. Lauri, J., 2022. If you had a choice, would you select speed over safety?. [Blog] *Futurice*, Available at: <https://futurice.com/blog/if-you-had-a-choice-would-you-select-speed-over-safety> [Accessed 6 October 2022].
13. Mustafa, T., 2022. *New safety app helps you find the best route home*. [online] Metro. Available at: <https://metro.co.uk/2022/01/18/new-safety-app-helps-you-find-the-best-route-home-15932866/#top> [Accessed 6 October 2022].
14. n.d. *Breadth First Search (BFS)*. [image] Available at: <https://vivadifferences.com/difference-between-dfs-and-bfs-in-artificial-intelligence/> [Accessed 6 October 2022].
15. NIKHIL KRISHNA, 1 J, 1 ML. Dijkstra complexity analysis using adjacency list and priority queue? [online]. Computer Science Stack Exchange. 2019. Available at: <https://cs.stackexchange.com/questions/104566/dijkstra-complexity-analysis-using-adjacency-list-and-priority-queue/> [Accessed 26 October 2022]
16. Pareek, S., 2015. *An App that Helps You Decide the Safest Route Home*. [online] The Better India. Available at: <https://www.thebetterindia.com/22908/app-helps-decide-safest-route-home/> [Accessed 6 October 2022].
17. Quora. 2022. *What routing algorithms are most likely being used by navigation services like Garmin and TomTom? What factors are included in the final ...*. [online] Available at: <https://www.quora.com/What-routing-algorithms-are-most-likely-being-used-by-navigation-services-like-Garmin-and-TomTom-What-factors-are-included-in-the-final-algorithm> [Accessed 6 October 2022].
18. Talukder, S., 2022. *Bellman-Ford Algorithm*. [image] Available at: <https://slidetodoc.com/bellmanford-algorithm-csci-385-data-structures-analysis-of/> [Accessed 6 October 2022].
19. Tiempo, C., 2022. *El 90,1 por ciento de las mujeres no denuncia el acoso callejero*. [online] El Tiempo. Available at: <https://www.eltiempo.com/colombia/medellin/el-90-1-por-ciento-de-las-mujeres-no-denuncia-el-

acoso-callejero-en-medellin-355056> [Accessed 6 October 2022].