Find the quickest and safest route to Medellin's streets

Manuel Villegas
Universidad Eafit
Colombia

Mvillegas6@eafit.edu.com

Esteban Muriel
Universidad Eafit
Colombia
Emurielr@eafit.edu.co

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

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

ABSTRACT

Sexual harassment on the street includes all the different actions such as gestures, unwanted comments, and other acts that lack respect for someone, especially these days, are the women who suffer the most from this phenomenon. Medellín isn't the first one and neither the only one to suffer from this, other cities like Mumbai in India create a solution to reduce sexual harassment to find the safest route to a certain location.

In the case of this project, it was used the Dijkstra algorithm to calculate the fastest and safest path from source coordination to a destination. For it, we related the distance and the value of sexual harassment at the destination in 3 different ways (Distance * harassment, Distance + (80*harassment), length^(2*harassment)). Also, hash tables and tuples were used to store the information of the streets of Medellin (distance, harassment, etc) because those are data structures very efficient in terms of time and memory, besides it makes it easy to access the information. Finally, the results were as we expected, with response times very fast and low complexity in memory

Key words

Shortest route, street sexual harassment, identificati of safe routes, crime prevention

1. INTRODUCTION

Sexual harassment on the street is one of many problems that humanity couldn't solve, so we are going to try to design a solution for this problem, in this case, an algorithm to find the safest and the shortest path to a certain location here on Medellín. A solution to this problem must be found if we are to improve ourselves as a society and as a country and if all citizens are to feel comfortable and secure.

1.1. The problem

In Medellín Colombia, 60% of the women population don't think the city is safe enough for them, so it is important to solve this problem to make them feel safer, also solving this problem would make the economy and the tourism growth even more and would make Medellin an incredible city.

1.2 Solution

Our main objective this semester is to implement an algorithm that would reduce both the risk of harassment and the distance between two points. We'll use a number of data structures, such as graphs as an adjacency list or as an adjacency matrix, linked lists, and others, to develop Dijkstra's algorithm method in Python.

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 Preventing Sexual Harassment Through a Path Finding Algorithm Using Nearby Search

This was a similar case in Mumbai, India, where they used thermal charts to predict high-risk locations for incidents of sexual harassment. The creation of this map and the use of Google maps API helped create the algorithm to find the safest places (the risk was 0 to 4, being 0 due safer). The main goal was to determine the shortest route and also the safest, they use the Euclidean equation of distance and to ensure that the line did not touch any grid point they used Bresenham's line drawing method.
They give priority to the safest route, followed by the route closest to the destination.[1]

2.2 A Data Integration and Analysis System for Safe Route Planning

Aryan Guatpaa and Bhavye Khetan also made a research about finding the best route to reduce the cases of sexual harassment against women, find a route to be both safe, fast the present it appealingly. To determine the safety of a route they took into account the number of cases in the area and the people in it and other different aspects and the safety value is equivalent to the combination of all of the factors already mentioned (from negative infinity to 100). They use the 'Bottom Up' method to build functions from the bases using the aforementioned factors.

They used Dijkstra's algorithm (or Dijkstra's Shortest Path First algorithm, SPF algorithm) to find the shortest in the graph. They modified Dijkstra's algorithm to make it consistent with the route's safety value[2]

2.3 Safety-aware routing for motorized tourists based on open data and VGI

It was a work tested during the nights of Los Angles that consisted in present an application for finding routes avoiding relatively dangerous areas in this city. This work was based on taking the information of how dangerous some areas in the city are, with something called VGI, that means, volunteered geographic information and with this information create the safest and shortest routes for tourists and vehicle drivers.
For this work, they took into account 4 principal factors. First, the streetlights because they consider this is one of the indicators for more safty at night. Then, police stations, because they say people are not likely to commit a crime near a police station. The third factor was, where are located the highways in Los Angeles because these places have a high number of cars at high speeds, so these are dangerous places for robbers. Then, they extracted the crime data set for the last 30 days.
Finally, they used the IDW method with all the information they recollected before. This method consists in that each measured value has a weight that is inversely proportional to the distance to the estimated point values.[3]

2.4 A data integration and Analysis System for safe route planning

This article proposes a new model to identify the shortest and safest route with the lowest risk score. For this, they use updated crime and accident data available on DYC open data. Also uses machine learning algorithms to generate the risk score of a path. They also consider a lot of factors that affect the safety of a path that normally are not considered in the processes of creating a GPS with the safest routes. Finally, they use 3 algorithms: data processing mark algorithm, k-Mean, and k Nearest neighbor [4]

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

[1] https://www.openstreetmap.org/
[2] https://osmnx.readthedocs.io/

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]
.

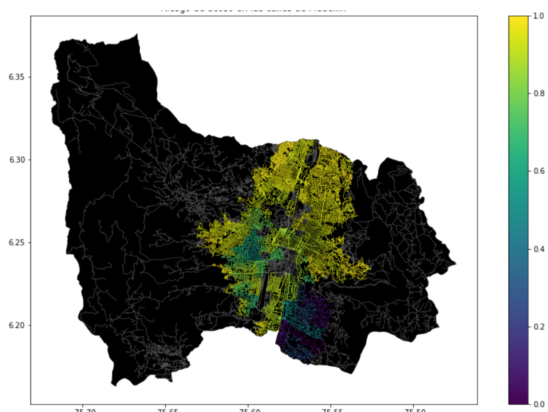


Figure 1. Risk of sexual harassment calculated as a linear combination of the fraction of households that feel unsafe and the fraction of households with income below one minimum wage, obtained from the 2017 Medellín Quality of Life Survey.

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

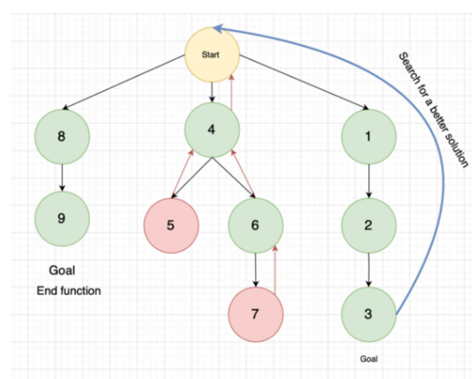
In the following, we present different algorithms used for a path that reduces both street sexual harassment and distance.

3.2.1 Backtracking algorithm

The first step is to validate every move. If we can't make any movement, we'll check the next one. To check this we follow the next order: if the upper cell is invalid, we check the lower cell. If the lower cell is also invalid we check for the right cell and if again this cell is invalid we finally check the left cell. In case all the moves are invalid, we backtrack to the last visited cell (This is done with a function that checks if the move is valid).

When we reach the end, we deliberately backtrack to explore any other possible path to reach the end.

Finally, after exhausting all possibilities, we output the best path to reach the end.
"The backtracking process of finding the shortest path in the maze is not efficient because it explores all possible paths to the destination, which may not be the final solution. " [Pencil Programmer, Anon. Complexity: O(K ^ N), where 'K' is the number of times the function calls itself. [5]



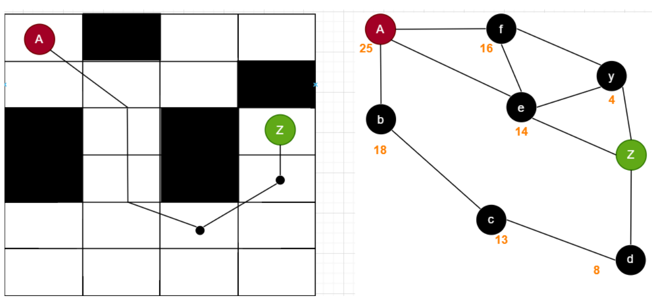
3.2.2 A* algorithm

3

https://github.com/VillegasMich/Project_Datos1.git

The A* algorithm is one of the best data structures to find a certain location on a maze, is using distance and heuristics to jump into different nodes and find the best path to the goal. Using heuristics can help to avoid different obstacles and plan ahead at each step to make a more optimal decision.

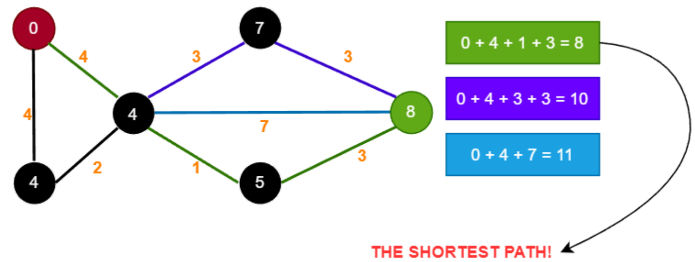
The algorithm focuses on making a low-cost path tree from the start node to the goal, thanks to the function f (n) =g (n) +h (n) where f (n) = total estimated cost of the path through the node n, g (n) = cost so far to reach node n, h (n) = estimated cost from n, n to the goal. This is the heuristic portion of the cost function, so it's like an estimation. It is important to have allowable heuristic values and the cost of each node in order for the function to function correctly. Complexity: O(bd) d = # nodes, b = branching factor [6]



3.2.3 Dijkstra Algorithm

This algorithm is designed to find the shortest path between two points in a chart. "Dijkstra's Algorithm works on the basis that any subpath B -> D of the shortest path A -> D between vertices A and D is also the shortest path between vertices B and D." this means that Dijkstra will find the shortest subpath of the points of the principal path.

Dijkstra over-estimates the distance of each point from the starting point, then visits each point from the sides and moves towards the nearest point. The time complexity of this algorithm is $O = (E \log log V)$, E = number of edges & V = number of vertices.[7]

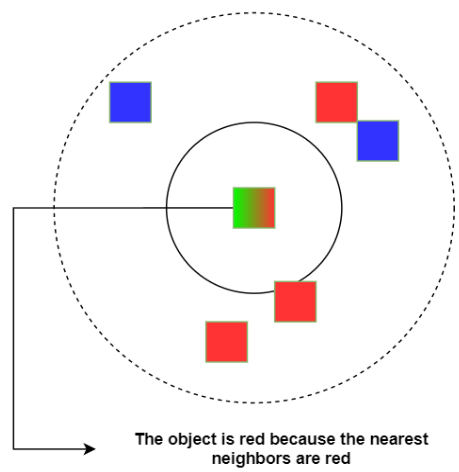


3.2.4 k-nearest neighbors algorithm

The KNN algorithm (k-nearest neighbors) is a simple machine learning algorithm useful to resolve classification and regression problems.

What it does is to work finding distances between a query and a specific number of examples(K) closest to the query.

We select the best K by trying several values of k and choosing the one that works better. Complexity: `O(k*n*d)` [8]

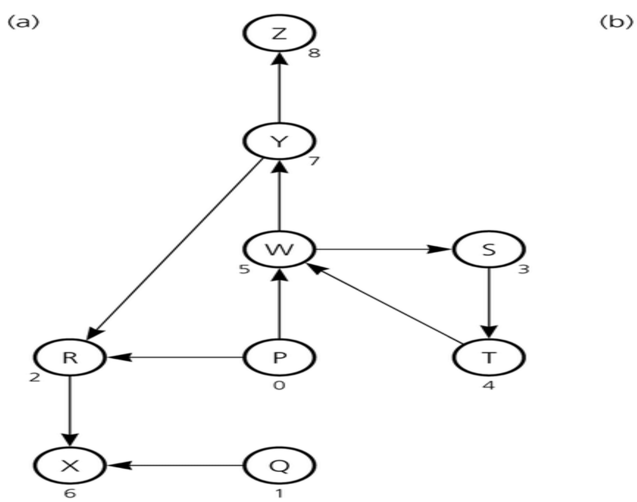


4. ALGORITHM DESIGN AND IMPLEMENTATION

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

For our representation of the map, we used an adjacency list using a dictionary, where each key is the node in the graph and the different values are other locations with the correspondent length and harassment values... The data structure is presented in Figure 2.



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 Algorithm for a pedestrian path that reduces both distance and risk of sexual street harassment

Dijkstra's algorithm:
This algorithm is essentially done with vertices or nodes **(u or v)** and weighted edges that are the ones that link 2 nodes. The edge is denoted (u, v) and the weight is represented by **w (u, v)**. The algorithm operates by initializing three values:

1. **dist**: It is an array that contains all the distances between the source node (s) and each node in the graph. The initialization way of the source node is **dist(s)=0** and the other nodes are initialized **dist(v)** = ∞. In the process of the algorithm are recalculated the values of **dist** from the source to each node v. And finally, the program is finalized when the shortest distance v is found.

2. **Q**: It is a queue of all the nodes in the graph. At the end of the process, the queue must be empty

3. **S**: It is an empty set to indicate which nodes have been visited. At the end of the process, **S** must contain all the nodes in the graph. [9]

. The algorithm is exemplified in Figure 3.

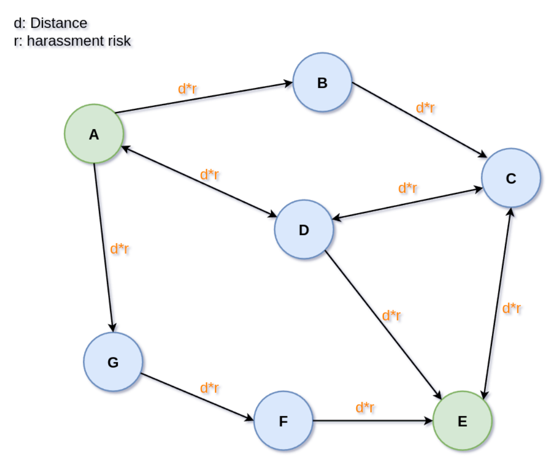
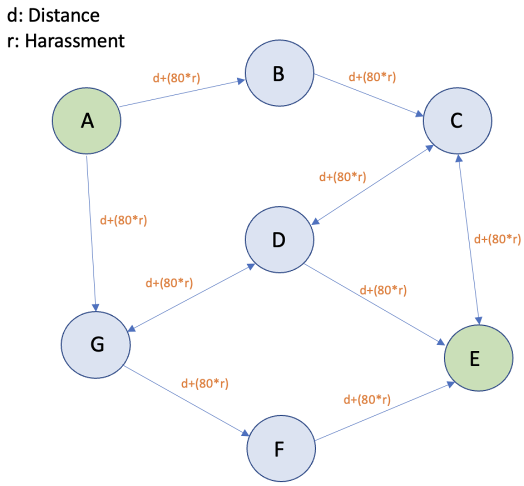


Figure 3: Calculation of a path that reduces both distance and risk of harassment

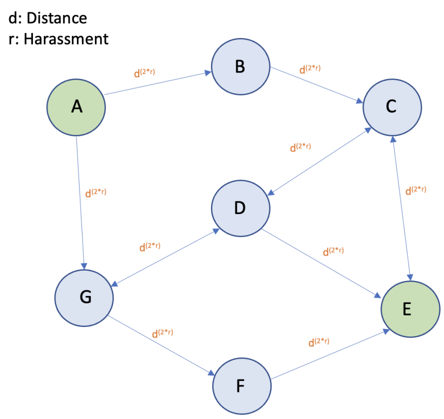
4.2.2 Calculation of two other paths to reduce both the distance and the risk of sexual street harassment

The second path was calculated by length+(80*harassment), and we noticed that in most of the cases that combination results in a straight path, which means the fastest route to travel. Also was the fastest path to being generated, so is very efficient

[4] https://github.com/VillegasMich/Project_Datos1.git



And finally, the third path was calculated by length$^{(2*harassment)}$. it resulted in a routes with many curves, so it isn't the fastest path to take



The algorithm is exemplified in Figure 4.

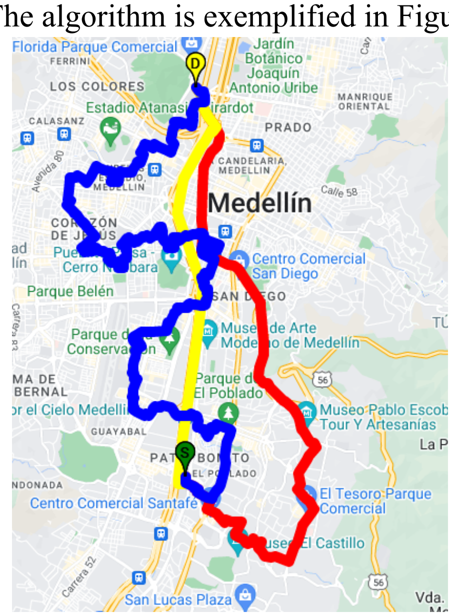


Figure 4: Map of the city of Medellín showing three pedestrian paths that reduce both the risk of sexual harassment and the distance in meters between the EAFIT University and the National University.

4.3 Algorithm complexity analysis

Algorithm	Time complexity
Dijkstra Algorithm	O(E * log $logV$)

Table 1: Time complexity of Dijkstra's algorithm using a priority queue where V represents the number of vertices and E represents the number of edges.

Data Structure	Complexity of memory
Hash table (dictionary)	O(V)

Table 2: Memory complexity of the Hash table or dictionary, where V is the number of keys.

4.4 Algorithm design criteria

We decided to use the Dijkstra algorithm because after researching we found that this algorithm was very appropriate and efficient to find the fastest route between nodes, so it is very useful when we have to work with maps as in the case of this project. We noticed that this was true because after implementing this algorithm in the project we noticed that it works very fast. Also, it is an algorithm that is relatively easy to implement, so it was very appropriate for us in this project.
Even though many algorithms can work similarly, we found that those algorithms can be better in other cases or work faster with fewer nodes, so as the project required to use a lot of nodes, we considered that Dijkstra would be better in this case.
In terms of the data structure, we decided to store all the information using hash tables, where we first use the source coordinate as the key, and the value is another hash table containing the destination coordinates as the key and a tuple

with the distance and the harassment as the value of this second hash table. We did it like this because hash tables are very efficient in terms of memory and time, besides is a data structure where is very easy to access the information

5. RESULTS

In this section, we present some quantitative results on the three pathways that reduce both the distance and the risk of sexual street harassment.

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

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

Origin	Destination	Distance (m)	Risk
Eafit	Unal	14571.9	0.0019
Eafit	Unal	8098.18	0.5566
Eafit	Unal	25527.69	0.73476

Distance in meters and risk of sexual street harassment (between 0 and 1) to walk from EAFIT University to the National University.

5.2 Algorithm execution times

In Table 4, we explain the ratio of the average execution times of the queries presented in Table 3.

Calculation of v	Average run times (s)
Dijkstra v = Length*Harassment	0,0937 s
Dijkstra v = Length + (80 * Harassment)	0,0748 s
Dijkstra v = length$^{(10*\text{Harassment})}$	0,1588 s

Table 4: Dijkstra algorithm execution times for each of the three calculator paths between EAFIT and Universidad Nacional.

6. CONCLUSIONS

As a conclusion, we could say that famous applications of maps like google maps or Waze should implement something similar to what we did in this project because always our security should be before how much time we take to arrive at a place, especially in cities like Medellin that can be dangerous in some places and streets. Also, we conclude that the Dijkstra algorithm works very well in this type of project, and this program could be used in a real situation because it works fast and doesn't consume a lot of memory. finally, we noticed that how we related the distance and the harassment is very important because the 3 paths are completely different, so it would be good to make an investigation to find which is the best way to relate these values

6.1 Future work

In the future, we would like to improve the project by implementing this algorithm in a mobile application or a web application that everybody could use. Also, it would be nice to use machine learning to allow the application to continue improving with time.

ACKNOWLEDGEMENTS

We are grateful for all the help we get from all the monitors, especially to Samuel Rico Gomez and Gregorio Bermudez Ocampo for comments that greatly improved this project. Thanks to all my companions and friends, also to Rafael Villegas Michel and Isabela Muriel Roldan for giving his opinion and tips to achieve this work in the best way possible. Finally thanks to our parents for giving us the opportunity to be in EAFIT.
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. Ma, D. 2022. Preventing Sexual Harassment Through a Path Finding Algorithm Using Nearby Search. *Omdena | Building AI Solutions for Real-World*

Problems.
https://omdena.com/blog/path-finding-algorithm/, August 9, 2022.

2. Guatpaa, A. and Khetan, B., 2020. Abstract of A Data Integration and Analysis System f - Count. *Ijsr.net*.

3. Keler, A. and Damascene, J., 2016. *Safety-aware routing for motorised tourists basedon open data and VGI*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/301566519_Safety-aware_routing_for_motorised_tourists_based_on_open_data_and_VGI.

4. Sarraf, R. and P., M., 2018. A Data Integration and Analysis System for Safe Route Planning. *Proquest.com*.

5. Anon, Shortest Path in Maze using Backtracking – Pencil Programmer. *Pencil Programmer*.

6. Abiy, T. and Pang, H., 2016. *A* Search | Brilliant Math & Science Wiki*. [online] Brilliant.org. Available at: https://brilliant.org/wiki/a-star-search/.

7. Dijkstra's Algorithm. 2022. *Programiz.com*. https://www.programiz.com/dsa/dijkstra-algorithm

8. Harrison, O., 2018. Machine Learning Basics with the K-Nearest Neighbors Algorithm. *Medium*.

9. Abiy, T., Pang, H. and Williams, C., 2022. *Dijkstra's Shortest Path Algorithm | Brilliant Math & Science Wiki*. [online] Brilliant.org. Available at: <https://brilliant.org/wiki/dijkstras-short-path-finder/>