Road network: Modeling and vulnerability analysis

Cristian Jimenez, Jaider Pinto, Jimmy Prieto

Abstract— In this paper, we conducted a literature review and collected data from the INVÍAS website on the main roads in Colombia. We organized and processed this data using Networkx, and applied measures of centrality to generate a network visualization. Our analysis revealed that Bogotá is the principal city in the country. By using the processed data, we can provide recommendations, such as identifying areas where public investments should be focused or identifying critical points in the network. Also, we used Pathfinding algorithms and community detection algorithms for further anazalysis.

Index Terms— **Network science, Road network, Road system, network centrality, CNA, .**

I. Introduction

The study also explores Colombia's challenges in developing transportation infrastructure and the vital role of transportation systems in economic growth. Additionally, complex network analysis is utilized to identify vulnerable nodes, connectivity issues, and clusters of related entities. The dataset used includes significant points and various sections of national roads represented as nodes and road segments as edges, forming a network model to evaluate the vulnerability of the road system. Also, we used Pathfinding algorithms and community detection algorithms gets some analysis plus.

II. Goal And Scope

Vulnerability analysis and possible effects on a logistics network In Colombia.

III. Literature Review

A. Colombian Road Network

The road network in Colombia is composed of the Primary Network (major highways managed by the national government), the Secondary Network (managed by departments), and the Tertiary Network (composed of inter-veredal roads or paths managed by municipalities). Colombia has a road network of 206,102 km, of which 6.9%, or 16,983 km, correspond to the Primary Network, 21% or 44,400 km correspond to the Secondary Network, and 69.46%, or 142,284 km correspond to the Tertiary Network.[1]

The Fourth Generation (4G) concession projects are the most ambitious road projects in the history of Colombia. Under a cost of approximately $18 billion in the Public-Private Partnership modality, 8,000 kilometers will be built, including 1,370 km of double-lane highways, and 160 tunnels in more than 40 new concessions. This will improve relevant aspects such as travel times, social and economic benefits, benefits for producers, road safety, connectivity, maintenance of the concession section, especially for freight, from manufacturing points to export ports, and accessibility.

It is projected that the works will be executed within a maximum of 6 years from the date of their award. When the 4G road construction is completed, one of the benefits of the project is expected to be a 30% reduction in travel distances.

For Colombia, this is a national point of interest, considering that land transport has high costs and long travel times. Therefore, strategies must be generated that contribute to achieving international competitiveness and infrastructure standards. Regarding travel times, approximately 80% of cargo transport in Colombia is carried out by land, through the country's six main logistics corridors: Bogotá-Buenaventura, Medellín-Villavicencio, Bogotá-Cúcuta-Caribe, Bogotá-Caribe, Rumichaca-Caribe, and Medellín-Cúcuta.

The road structure in Colombia presents permanent challenges for several reasons, including the country's structural and topographical conditions not being the best, as Colombia has three mountain ranges that generate gaps in connectivity for the country.[2]

B. Transport infrastructure

The development of transportation infrastructure is crucial for the economic growth of a nation. Good transportation systems contribute to the redistribution of economic activities and the development of prosperous regions, requiring continual transportation improvements. Many studies have found a positive correlation between transportation and economic development. Accessibility to cities is particularly significant for their competitive advantage and tourism industry. Ground transportation is viewed as critical infrastructure, essential for basic societal operations, and has environmental benefits over road-only transportation.

Various studies have investigated the importance of the transportation system for the economy, society, and the environment using different methods. One such method is Complex Network Analysis (CNA), which represents the interactions between system entities using links (represented by nodes in the network model, such as cities or stations in transportation studies). A topological analysis is conducted on the network structure to identify vulnerable nodes, connectivity issues, or clusters of related entities. CNA has been used in many countries to study different modes of transportation, including air transportation, subways, buses, railways, and maritime transportation.[3]–[5]

IV. Case Study

In our project we want to review the different metrics and measures that we can apply to our dataset on the different nodes and road routes in our country and review the feasibility of performing vulnerability analysis on the resulting network.

V. Data Set

The data set used in this article was found in the 0 website, they showed enter to finish city road, all the data is bidirectional and is distributed throughout the country. The dataset was an excel so we had to make some process of the data to get our dataset for the network.

VI. Implemented Network Science Approach

To carry out the desired approach and development, we began with obtaining data as detailed below with the data life cycle, and then, with the data obtained, we transformed this data into a network (model explain as a complex network) in which we primarily obtained and analyzed centrality measures.

A. Data Life Cycle

1) Generation

The data generation was carried out by the National Institute of roadways (INVIAS) through a data collection over the years, mapping the different national routes in the country. This data contains each of the important points and different sections of each of the national roads in the country.

2) Collection

This stage of the cycle was carried out by us. Using the various open data provided by INVIAS, we collected the most important data from the national routes, taking the origin and destination points of each section as the basis for the data we needed.

3) Processing

The data processing consisted of 2 important parts. The first was the cleaning of the dataset, in which irrelevant points as well as incomplete or erroneous data were eliminated. Then, the obtaining of origin and destination points was carried out so that pairs of nodes were available. This had to be done because there were sections that included more than 2 cities, so these cases had to be divided to obtain a clean and structured dataset.

4) Storage

With the cleaned dataset, we saved only the necessary data, which was stored in a table with two columns (currently): the origin and destination. Each column contains the name of the node (city, town, point of interest, etc.).

5) Management

This data was used to generate a network in which the origin and destination points are the nodes of the network, and the edges represent the route between 2 nodes. This network is our model with which we will start the analysis of centrality metrics and propose a vulnerability analysis of the road network.

6) Analysis

We obtained the different centrality measures of the road network to perform an analysis of the most important points of the network, as well as the most vulnerable ones. Next, we make a prediction of possible critical points and predict viable options for preventing network problems.

7) Visualization

In this step we model the red using NetworkX and use this for show the resulting network. At the beginning we obtain a network in which a lot of nodes was disconnected. After that, corrections in data were made and we obtained a good network in which just exist 2 nodes out of the main component a very accurate result according to the Colombian vial network.

8) Interpretation

With the visualization of the network and the before analysis we can detect de main nodes and propose an interpretation about how some nodes are critical and if these nodes disappear a lot of main cities will be disconnected.

B. Model of complex network

Complex networks are simplified representations of complex real-world systems using graph theory, which capture the intricate interactions and connections within these systems. A network is made up of nodes and edges, where each node represents an individual entity in the system and each edge represents the relationship or interaction between two entities. As the road system is a constantly evolving complex system, it can also be analyzed using complex network theory. In this study, intersections are considered as nodes, and road segments are represented as edges to construct a network model for analyzing the vulnerability of road system. Therefore, the road network can be represented as a graph $\mathrm{G} = \{\mathrm{V}, \mathrm{E}, \mathrm{W}\}$, where $V = \{v_i | i = 1,2,3, \dots, N\}$ is the set of nodes, $E = \{e_{ij} | e_{ij} = (v_i, v_j)\}$ is the set of edges, $W = \{\mathrm{w}_{\mathrm{ij}} | w_{ij}(v_i, v_j)\}$ is the set of edge weights, and N is the number of nodes.[3], [5]

C. Centrality Measures

Freeman's significant contributions to structural sociology involved consolidating and reviewing previous research dating back to the early fifties, including works by Bavelas, Leavitt, Shimbel, and Shaw. Freeman's work established the first set of centrality indices, including degree, closeness, and betweenness centrality. The fundamental concept in structural sociology is to represent a social or organizational group as a network with nodes representing individuals and edges representing their relationships. Bavelas was the first to recognize that central individuals in a social network often hold a prominent role in the group, with a good location in the network structure correlating with independence, influence, and control over others. The same idea can be used for another kind of networks like in this case the road network, these measures can help to understand the importance of a node (city or main place) in the network and the isolation of each node making see the importance of each node and how the existence/inexistence can affect the network.[5]

1) Degree Centrality

The concept of degree centrality is based on the notion that important nodes within a graph have the most connections to other nodes. The degree of a node corresponds to the number of edges associated with the node, which is equivalent to the number of its immediate neighbors. The normalization used for degree centrality produces values between 0 and 1, with a value of 1 indicating that a node is connected to every other node within the graph. However, degree centrality may not be particularly relevant in primal urban networks where a node's degree (the number of roads connected to that node) is constrained by geographic factors.[5]The degree k_i of node i is defined in terms of the adjacency matrix as:

$$k_i = \sum_{j \in N} a_{ij} \qquad (1)$$

The degree centrality (C^D) of i is defined as:

$$C_i^D = \frac{k_i}{N-1} = \frac{\sum_{j \in N} a_{ij}}{N-1} \qquad (2)$$

2) Closeness Centrality

The most basic interpretation of closeness is founded on the notion of minimum distance or geodesic d_{ij}. This represents the shortest possible summation of edge lengths over all feasible paths within a weighted graph between i and j, or in a topological graph, the minimum number of edges traveled. C^C, or closeness centrality, is best utilized when evaluating measures that rely on independence. This centrality index is only applicable to connected graphs unless one assumes a finite value for d_{ij} when there is no available path between two nodes i and j.[5] For non-valued graphs, the C^C centrality index ranges from 0 to 1. The closeness centrality (C^C) of i is defined as:

$$C^C = \frac{N-1}{\sum_{j \in N, j \neq i} d_{ij}} \qquad (3)$$

3) Betweenness Centrality

The relationships between two points that are not directly connected may be influenced by other actors, particularly those along the paths between them. Therefore, actors in the middle may exert strategic control and influence over the others. The basic concept of betweenness centrality is that an actor is central if it lies on many of the shortest paths connecting other actors. Namely, if n_{jk} is the number of geodesics linking the two actors j and k, and $n_{jk}(i)$ is the number of geodesics linking the two actors j and k that contain point i, the betweenness centrality of actor i can be defined as:

$$C_i^B = \frac{1}{(N-1)(N-2)} \cdot \sum_{\substack{j,k \in N \\ j \neq k; j,k \neq i}} \frac{n_{jk}(i)}{n_{jk}} \qquad (4)$$

C_i^B is a measure of betweenness centrality that ranges between 0 and 1 and is highest when actor i lies on all the shortest paths. Freeman has proposed various extensions to the betweenness index. When communication does not necessarily follow the shortest path, a more realistic betweenness measure should consider non-shortest paths as well. Two such measures are flow betweenness and random path betweenness, but in our study, we focus only on the simplest case, which is the shortest path betweenness described in formula (4)[5].

D. Pathfinding algorithms

Pathfinding algorithms are commonly employed to determine the most efficient path between two nodes in a network, with some algorithms aiming to establish the minimum number of connections or the total weight of these connections to link a given network. These algorithms are widely applied in transportation, communication, and logistics networks to identify the best routes[6]

In the context of a graph G and given two vertices u and v belonging to V, a path P between u and v is an ordered sequence of edges, represented by N, of the form.

$$P = \{(u, v_1), (v_1, v_2), \ldots, (v_{N-2}, v_{N-1}), (v_{N-1}, v)\}$$

It's important to note that there can be multiple paths between two nodes, and therefore, paths are not unique.

We define two nodes u and v as connected if there exists a path P between them. Additionally, given a path P between two nodes u and v belonging to V, and a node n present in the path, we say that n_0 is the successor of n if (n, n_0) belongs to P. In other words, n_0 is the next node that would be visited when following the path P after reaching n.

The optimal path between two nodes u and v is defined as a path P in the set of all existing paths between u and v, denoted by $\chi(u, v)$, where the distance of P, represented by d(P), is less than or equal to the distance of any other path in $\chi(u, v)$. In other words, P is optimal if d(P) is the smallest distance among all paths connecting u and v. If a path P is optimal, then it is also considered the shortest distance path, and we denote the distance of this path as $\delta(u, v)$. It's worth noting that in a finite graph, an optimal path between any two nodes always exists since the set $\chi(u, v)$ is finite in this case.

1) BFS

Breadth-First Search, abbreviated as BFS, is a methodical algorithm that attempts to find a path by examining all the neighbors of each node it visits. The algorithm maintains a queue to keep track of the next nodes to be examined. It adds all unvisited neighbors of a node when it is examined until the queue is empty. To avoid exploring the same node twice, the algorithm stores the explored nodes in a set. The set of nodes that haven't been explored yet is often called the open set, while the set of visited nodes is referred to as the closed set. Nodes in the open set are called open nodes, while nodes in the closed set are known as closed nodes.[7]

```
procedure BFS($G, \alpha, \beta$)
Input: Graph $G = (V, E)$, directed or undirected; source node $\alpha \in V$; goal node $\beta \in V$
Output: Given $u \in V$, previous[$u$] gives us the node come from to reach $u$ in the path
    computed
    Q ← $Queue()$
    $S$ ← $Set()$                                    ▷ Keeps track of explored nodes
    previous ← $Map()$
    for $u \in V$ do
        previous[$u$] ← $\varnothing$
    end for
    Q.enqueue($\alpha$)
    $S$.add($\alpha$)
    while not Q.empty() do
        $u$ ← Q.front()
        Q.dequeue()
        for v $\in u$.neighbours() do
            if $v \notin S$ then
                Q.enqueue($v$)
                S.add($v$)
                previous[$v$] ← $u$
            end if
        end for
    end while
    return ReconstructPath(previous, $\alpha$, $\beta$)
end procedure
```

Fig 1. BFS algorithm[7]

```
Algorithm 2 Reconstruct path
procedure ReconstructPath(previous, $\alpha$, $\beta$)
Input: The map previous returned by the pathfinding algorithm; source node $\alpha \in V$; goal
    node $\beta \in V$
Output: An ordered list $P$ with the nodes from the path from $\alpha$ to $\beta$
    $P$ ← []                                    ▷ Empty array for the path
    $u$ ← $\beta$
    while $u \neq \alpha$ do
        $P$.push($u$)
        $u$ ← previous[$\beta$]
    end while
    $P$.push($\alpha$)
    return $P$.reversed()
end procedure
```

Fig 2. Reconstruct path algorithm[7]

2) Dijkstra

Dijkstra's algorithm is another well-known algorithm in graph theory used to find optimal paths between nodes in a graph with positive weights. Instead of a simple queue, the algorithm uses a priority queue to explore all unvisited neighbors of a node. The algorithm inserts the neighbors into the priority queue based on the distance of the edge so that the node with the least distance from the source is extracted first.

Typically, the algorithm only takes a starting node and computes optimal paths to all reachable nodes from the origin. However, since we only need to find the path to one of the goal nodes, we use the early exit condition to terminate the execution as soon as we expand a goal node. This still provides us with the optimal path.[7]

```
procedure Dijkstra($G, \alpha, T$)
Input: Graph $G = (V, E)$, directed or undirected; source node $\alpha \in V$; set of goal nodes
    $T \subset V$
Output: Given $u \in V$, previous[$u$] gives us the node come from to reach $u$ in the path
    computed, and d[$u$] gives us the distance to that node, if it has been explored
    Q ← $PriorityQueue()$
    $S$ ← $Set()$
    d ← $Map()$                     ▷ Keeps track of the shortest distance to each node
    previous ← $Map()$
    for $u \in V$ do
        d[$u$] ← $\infty$
        previous[$u$] ← $\varnothing$
    end for
    Q.insert((0, $\alpha$))
    d[$\alpha$] ← 0
    while not Q.empty() do
        u ← Q.removeMin()
        S.add(u)
        if $u \in T$ then                                  ▷ Early exit condition
            return ReconstructPath(previous,$\alpha$,$u$)
        end if
        for $v \in u$.neighbours() do
            if $v \in S$ then
                continue                              ▷ Already explored node
            end if
            alt ← d[$u$] +$w((u,v))$
            if alt < d[$v$] then
                d[$v$] ← alt
                Q.insert(($v$, alt))
                previous[$v$] ← $u$
            end if
        end for
    end while
end procedure
```

Fig 3. Dijkstra's algorithm[7]

3) Greedy Best-First

Greedy Best-First search is an algorithm that finds a path in a graph using a heuristic function. The heuristic function provides an estimate of the distance from a node to one of its preferred goal nodes, which helps to determine the order in which nodes are explored. The algorithm explores the node closest to the goal first, not necessarily the one with the shortest distance travelled so far. This approach can lead to faster execution times compared to Dijkstra's algorithm, but it doesn't always produce the optimal path. The heuristic function doesn't consider the actual costs of travelling between nodes, so the algorithm may not find the shortest path in some cases.[7]

```
procedure GreedyBestFirstSearch($G, \alpha, T$)
Input: Graph $G = (V, E)$, directed or undirected; source node $\alpha \in V$; set of goal nodes
    $T \subset V$
Output: Given $u \in V$, previous[$u$] gives us the node come from to reach $u$ in the path
    computed, if it has been explored
    Q ← $PriorityQueue()$
    $S$ ← $Set()$
    previous ← $Map()$
    for $u \in V$ do
        previous[$u$] ← $\varnothing$
    end for
    Q.insert((0, $\alpha$))
    while not Q.empty() do
        $u$ ← queue.removeMin()
        S.add($u$)
        if $u \in T$ then                                  ▷ Early exit condition
            return ReconstructPath(previous,$\alpha$,$u$)
        end if
        for $v \in u$.neighbours() do
            if $v \in S$ then
                continue
            end if
            Q.insert(($v$, $h(v)$))
            previous[$v$] ← $u$
        end for
    end while
end procedure
```

Fig 4. Greedy Best-First search algorithm[7]

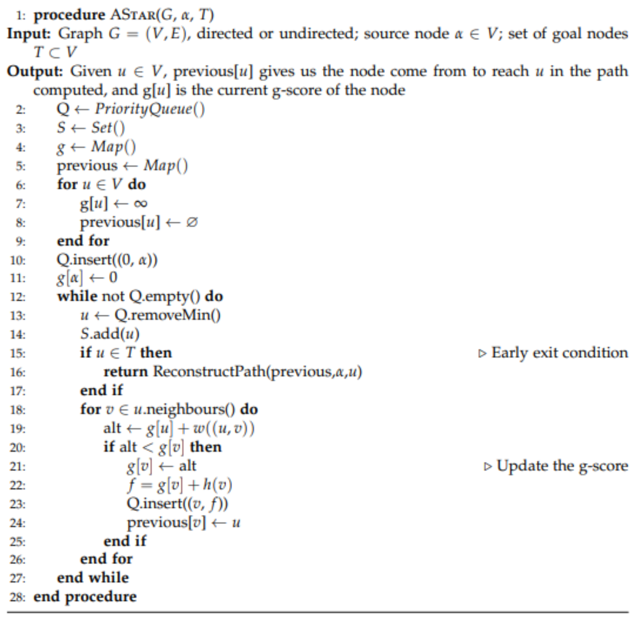
*4) A**

The A* algorithm uses a heuristic function to estimate the distance from a node to the goal node(s), but it also considers the actual distance travelled from the source node to the current node. The heuristic function used in A* must satisfy the admissibility condition, which means that it never overestimates the distance to the goal. In other words, the estimated distance from a node to the goal must always be less than or equal to the actual shortest distance to the goal.

The A* algorithm uses a priority queue to explore nodes in the order of their estimated total cost, which is the sum of the actual distance travelled from the source to the current node and the estimated distance from the current node to the goal. It starts with the source node and continues exploring the neighboring nodes with the lowest estimated total cost, until it reaches a goal node or until all nodes have been explored.

If a path is found to a goal node, it is guaranteed to be optimal, because the algorithm explores nodes in a way that minimizes the estimated total cost. However, if the heuristic function is not admissible, the algorithm may explore more nodes than necessary, which can slow down the search.

A* is a widely used pathfinding algorithm in many applications, such as robotics, video games, and route planning, due to its efficiency and reliability.



```
procedure ASTAR($G$, $\alpha$, $T$)
Input: Graph $G = (V, E)$, directed or undirected; source node $\alpha \in V$; set of goal nodes
    $T \subset V$
Output: Given $u \in V$, previous[$u$] gives us the node come from to reach $u$ in the path
    computed, and g[$u$] is the current g-score of the node
    Q ← $PriorityQueue()$
    $S \leftarrow Set()$
    $g \leftarrow Map()$
    previous ← $Map()$
    for $u \in V$ do
        g[$u$] ← $\infty$
        previous[$u$] ← $\varnothing$
    end for
    Q.insert((0, $\alpha$))
    $g[\alpha] \leftarrow 0$
    while not Q.empty() do
        $u$ ← Q.removeMin()
        $S$.add($u$)
        if $u \in T$ then                                  ▷ Early exit condition
            return ReconstructPath(previous,$\alpha$,$u$)
        end if
        for $v \in u$.neighbours() do
            alt ← $g[u] + w((u, v))$
            if alt < $g[v]$ then
                $g[v]$ ← alt                                ▷ Update the g-score
                $f = g[v] + h(v)$
                Q.insert(($v$, $f$))
                previous[$v$] ← $u$
            end if
        end for
    end while
end procedure
```

Fig 5. A algorithm*[7]

E. Community detection

The concept of community detection has emerged in network science as a method for finding groups within complex systems through represented on a graph. In contrast to more traditional decomposition methods which seek a strict block diagonal or block triangular structure, community detection methods find subnetworks with statistically significantly more links between nodes in the same group than nodes in different groups (Girvan and Newman, 2002). Central to community detection is the notion of modularity, a metric that captures this difference [8]:

$$Q = \frac{1}{2m}\sum_{i,j} A_{ij} - \frac{k_i k_j}{2m}\zeta g_i g_j$$

When analyzing different networks, it may be important to discover communities inside them. Community detection techniques are useful for social media algorithms to discover people with common interests and keep them tightly connected. Community detection can be used in machine learning to detect groups with similar properties and extract groups for various reasons. For example, this technique can be used to discover manipulative groups inside a social network or a stock market.

VII. Preliminary results

First, we must process the data and then do several steps to get the information required to create a network using Networkx. The preliminary network that we obtain is shown in the next figure:

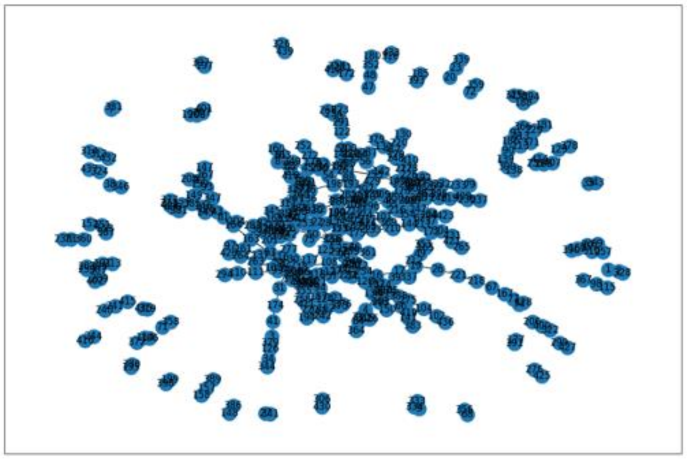


Fig 6. Preliminary Network

From the obtained network we can see that it is not connected, there are many nodes that are isolated and connected with a few other nodes, but without being connected to our large component.

We apply some of the centrality measures for a network and see that the node 6, has the largest value in degree centrality:

6: 0.02968036529680365, the largest value in closeness centrality 6: 0.07820045333596137 and the largest value in betweenness centrality 6: 0.23994774508238034, this is expected since node 6 corresponds to Bogota, which is the largest and most important city in the country, so it gives meaning to our network.

We also have gotten the great component from our network of which we are going to carry out the vulnerability validations:

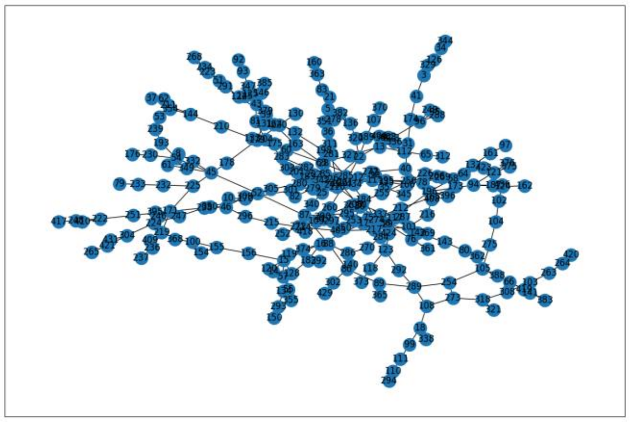


Fig 7. Preliminary Main Network component

VIII. Final Results

Finally, we had to make several improvements to the initial nodes obtained, connecting them in a logical way, to achieve a great connected component, the Leticia and Tarapaca nodes were simply disconnected, because they are not connected to the giant component. The giant component has 413 nodes and 432 edges.

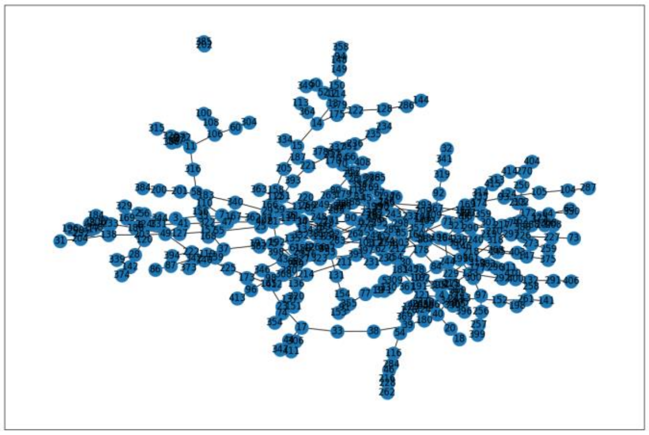


Fig 8. Fixed Network

And then we began to perform corresponding analyzes on this giant component.

Initially we apply the measure of average shortest path to get an initial idea of our network. This result 16.29 show us that it is not a small world network. We then started running reviews of network centrality measures to see if it was consistent.

Nodo	Clossness Centrality
6-Bogota	0,09592549476
274-Villeta	0,09365764947
30-Honda	0,09325486646
271-Tocancipa	0,0923145866
191-Mariquita	0,0916981972

As for closeness centrality, we find consistent values, there is Bogota, Villeta, Tocancipa are quite central cities on a map, there is also Honda and Mariquita, connection cities between Bogota and the west of the apis, (Medellin, Cali)

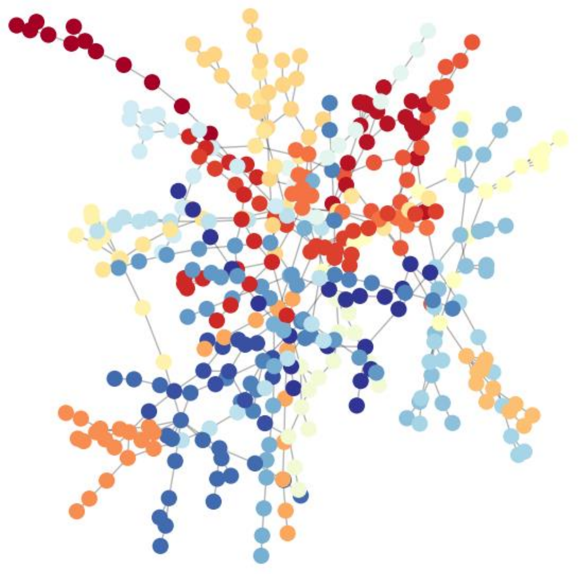
Nodo	Degree Centrality
6-Bogota	0,02912621359
164-Popayan	0,02184466019
121-Rionegro	0,02184466019
11-Cucuta	0,01699029126
37-San Gil	0,0145631068

In the previous table we see the cities with the Degree Centraily.

Nodo	Betweenness Centrality
6-Bogota	0,5222316711
34-Choconta	0,3229631729
271-Tocancipa	0,3074768305
189-Tunja	0,2984728226
30-Honda	0,2933440421

Finally, we obtained the Betweenness Centrality metrics of the nodes with the highest value, reviewing the general results, we see that Bogota is emerging in our network as the most important city, having the highest value in these 3 metrics. What gives meaning to the network since Bogota is effectively the most important city in the country. Two nodes (Cities) that also appear with a good value of closeness centrality and Betweenness centrality are Honda, and Tocancipa.

We also tried to get the communities using the Louvain algorithm, which finds the best partitions, but it found too many and the division between the communities is not so clear.

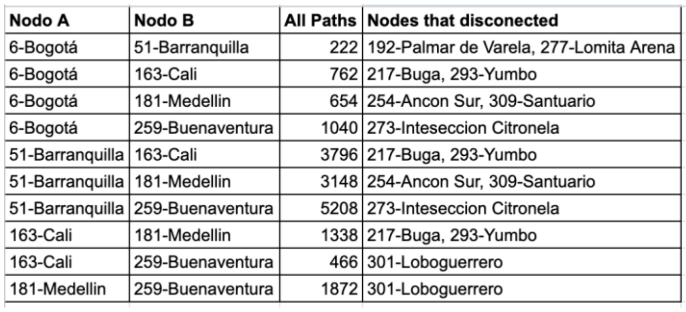


To finish our analysis we wanted to carry out a validation of connectivity between the main cities of the country, we chose the following cities due to their importance in the country: Bogota, Barranquilla, Cali, Medellin, Buenaventura (Buenaventura for being the largest port in the country in the Pacific), to see the number of paths that are created between them and which nodes would need to be removed to disconnect these two cities.

First, we obtain the shortest path between them:

Node A	Node B	Shortest_Path
51-Barranquilla	163-Cali	[51, 192, 81, 47, 41, 49, 222, 225, 165, 23, 17, 33, 38, 39, 40, 343, 278, 177, 268, 210, 388, 207, 208, 288, 217, 301, 163]
51-Barranquilla	259-Buenaventura	[51, 192, 81, 47, 41, 49, 222, 225, 165, 23, 17, 33, 38, 39, 40, 343, 278, 177, 268, 210, 388, 207, 208, 288, 217, 301, 297, 273, 259]
259-Buenaventura	181-Medellin	[259, 273, 297, 301, 163, 293, 185, 188, 249, 248, 255, 283, 103, 253, 254, 181]
6-Bogota	259-Buenaventura	[6, 274, 30, 191, 278, 177, 268, 210, 388, 207, 208, 288, 217, 301, 297, 273, 259]
6-Bogota	181-Medellin	[6, 45, 246, 243, 289, 253, 254, 181]
51-Barranquilla	181-Medellin	[51, 192, 81, 82, 80, 83, 282, 285, 117, 111, 189, 34, 271, 6, 45, 246, 243, 289, 253, 254, 181]
259-Buenaventura	163-Cali	[259, 273, 297, 301, 163]
6-Bogota	51-Barranquilla	[6, 271, 34, 189, 111, 117, 285, 282, 83, 80, 82, 81, 192, 51]
181-Medellin	163-Cali	[181, 254, 253, 103, 283, 255, 248, 249, 188, 185, 293, 163]
6-Bogota	163-Cali	[6, 274, 30, 191, 278, 177, 268, 210, 388, 207, 208, 288, 217, 301, 163]

We saw that it makes sense, for example relatively close cities like Cali and Buenaventura only need 4 edges and separate cities like Barranquilla and Buenaventura, if they require going through more than 20 nodes to connect.



Nodo A	Nodo B	All Paths	Nodes that disconected
6-Bogotá	51-Barranquilla	222	192-Palmar de Varela, 277-Lomita Arena
6-Bogotá	163-Cali	762	217-Buga, 293-Yumbo
6-Bogotá	181-Medellin	654	254-Ancon Sur, 309-Santuario
6-Bogotá	259-Buenaventura	1040	273-Inteseccion Citronela
51-Barranquilla	163-Cali	3796	217-Buga, 293-Yumbo
51-Barranquilla	181-Medellin	3148	254-Ancon Sur, 309-Santuario
51-Barranquilla	259-Buenaventura	5208	273-Inteseccion Citronela
163-Cali	181-Medellin	1338	217-Buga, 293-Yumbo
163-Cali	259-Buenaventura	466	301-Loboguerrero
181-Medellin	259-Buenaventura	1872	301-Loboguerrero

As you can see in the table above, even though there are a large number of paths between the nodes, it is only necessary to disconnect a few nodes, to prevent communication between the large and important nodes of the country, which effectively shows us the logistics network of the country. The country is susceptible to attacks, or some catastrophe, which are things that can happen in Colombia, we are still in an armed conflict, and we have a mountainous geography with constant landslides. The nodes that disconnected the roads between important cities are in the cities except Bogota. Bogota is very well connected to the network, so the affectation occurs above all when reaching other cities when the road is from Bogota.

IX. Links

A. *Source code*

https://github.com/Jaider0111/NSFDA_09_Term_Project

B. *Explicative video*

https://drive.google.com/file/d/1kcVVTaObs5HtrQdxrxHES2IFTgZBkBax/view?usp=share_link

X. Team Members

Team Member	Role	Activities
Jaider Pinto	Leader	Guide the team for the goal.
Cristian Jimenez	Investigator	Discover
Jimmy Prieto	Investigator	Apply

Table 1. Team members

XI. Conclusions

We see that the initially obtained network makes sense by placing expected nodes as important, as nodes with the highest metrics

There are many nodes, separated from the large component, we must carry out a major review to see if this is the case, (they are isolated) or it is a matter of the data.

Carrying out a manual arrangement of the data, connecting disconnected components, obtaining an interesting network with which we were able to work.

According to our review of removing nodes to eliminate communication between important nodes, the logistics network is vulnerable and only requires removing a maximum of two nodes to separate important cities.

Perhaps a review can be carried out with an expert to make the dataset more complete, and finally obtain more accurate results on this logistics network.

References

[1] "2.3 Colombia Red Carretera - Logistics Capacity Assessment - Digital Logistics Capacity Assessments." https://dlca.logcluster.org/display/public/DLCA/2.3+Colombia+Red+Carretera (accessed Apr. 17, 2023).

[2] "Portal INVÍAS - Colombia." https://www.invias.gov.co/ (accessed Apr. 17, 2023).

[3] Z. Wang, Y. Pei, J. Liu, and H. Liu, "Vulnerability analysis of urban road networks based on traffic situation," *International Journal of Critical Infrastructure Protection*, vol. 41, p. 100590, Jul. 2023, doi: 10.1016/J.IJCIP.2023.100590.

[4] L. Calzada-Infante, B. Adenso-Díaz, and S. García Carbajal, "Analysis of the European international railway network and passenger transfers," *Chaos Solitons Fractals*, vol. 141, Dec. 2020, doi: 10.1016/J.CHAOS.2020.110357.

[5] S. Porta, P. Crucitti, and V. Latora, "The network analysis of urban streets: A primal approach," *Environ Plann B Plann Des*, vol. 33, no. 5, pp. 705–725, 2006, doi: 10.1068/B32045.

[6] T. Bratanic, *Graphs and network science: An Introduction*.

[7] D. Monzonís Laparra, "PATHFINDING ALGORITHMS IN GRAPHS AND APPLICATIONS."

[8] A. Allman, W. Tang, and P. Daoutidis, "Towards a Generic Algorithm for Identifying High-Quality Decompositions of Optimization Problems," 2018, pp. 943–948. doi: 10.1016/B978-0-444-64241-7.50152-X.