EECS 4414 Project Final Report
Evaluation of Transit Systems' Robustness under Service Disruptions

Paul Kim
York University
pyjaekim@my.yorku.ca

Dennis Phetsomphou
York University
dennisph@my.yorku.ca

Abu Saeed
York University
aburocks@my.yorku.ca

Patrick Tan
York University
tanpatri@my.yorku.ca

1 ABSTRACT

In this study, the robustness of transit systems is evaluated under the context of service disruptions. Transit systems are not equally reliable in the presence of these disruptions. Therefore, determining how the design of the transit network affects its ability to handle them is important to measure robustness. Due to the complexity of this problem, several approximations, assumptions and constraints have been considered to help approach this study. The route-based network graphs are constructed by processing the General Transit Feed Specification (GTFS) data of transit systems. To provide a comprehensive snapshot of the robustness, three metrics are considered in response to disruptions: the in-paper defined "rerouting coefficient", the size of the giant strongly connected component (GSCC), and changes in edge betweenness centrality. Evaluation of these three metrics was performed by comparing transit systems in two different cities: Toronto Transit Commission (TTC) in Toronto and the Réseau de transport de la Capitale (RTC) in Quebec. Under the assumptions and constraints of this study, the TTC outperformed the RTC with regards to the robustness of the transit networks.

2 INTRODUCTION/MOTIVATION

For urban cities in particular with a crowded/dense population, it is crucial to a city's economy and the daily lives of the population to move as many people as efficiently possible from point A to point B. Given a dense population, in the extreme case if each adult owned a car and decided to drive their own vehicle, road traffic would unnecessarily be over congested with that many vehicles on the streets. Having a low passenger per vehicle ratio is simply non-optimal for improving road traffic. The hassle of having to be conscious and aware as one painstakingly drives at a slow pace through heavy traffic, the cost associated with driving your own vehicle (maintenance, gas, parking fees) and finding parking where it is often limited are typically enough reason to turn people away from driving in the city. Realistically, not all adults own cars or choose to drive their cars because it is often not practical to drive, assuming there is an alternative form of transportation that is reasonably reliable.

Transit systems help to alleviate road congestion by moving large volumes of people systematically (with high passenger per vehicle ratio) from one point to another on a regular basis. Transit systems normally operate with a combination of a metro/subway system and connected bus routes. By having strategically planned bus routes and service lines, transit systems have the potential to attract more ridership from the population and improve the average road traffic.

However, not all transit systems are made equally reliable for their respective city for a multitude of factors, and they may not necessarily be as well designed as they could be to handle service disruptions. These service disruptions can be obstacles or delays that can affect transit systems such as road/track maintenance, emergencies/accidents, weather conditions, severe ridership congestion or simply heavy road traffic. For example, an accident in the middle of the street or perhaps construction that closes off a part of the road can result in a block within the transit system in that direction between the two transit stops.

We are interested in seeing how well designed a transit system is in terms of its ability to handle service disruptions. Is the transit system still able to efficiently move passengers from stop/station A to stop/station B within a comparatively reasonable timeframe via other paths in the presence of disruptions? How connected is the transit system even after a series of disruptions? How does route importance diverge when that connection becomes unavailable due to a disruption? How does this unavailability affect all other connections? Which connection becomes the next most important? These are the questions we want to answer and analyze.

By deriving rigorous metrics from this problem through a methodical approach, we hope to precisely see what kind of network graph properties, trends and quantifiable measures that may contribute to a well designed transit system. This problem is important as it could help identify areas of weakness in any transit system and possibly improve considerations in urban planning [5]. While the main application here is towards a transit system, it can also be extended to all kinds of transportation networks in general where a roadblock or bottleneck could occur and the object of interest must be rerouted effectively.

3 PROBLEM DEFINITION

3.1 Network Implementation Options, Constraints and Restrictions

To approach the problem of evaluating the robustness of the transit system, we first need to translate the transit system into a simple network model that we can use to simulate disruptions. We classify and contrast two ways of approaching the construction of such a network.

3.1.0.1 Road Based Network. A road (infrastructure) disruption affects all service routes that contain the node/edge where a disruption occurs. At minimum, this can be modeled as an undirected simple graph and it requires road infrastructure data that can be mapped to a network model (intersections as nodes, road connections as edges with the distance/time as edge weights).

3.1.0.2 Route Based Network. A route disruption only affects the service route in which the disruption occurs even when there are collinear/overlapping portions of the path along the same road infrastructure. This requires a directed multi-graph that maps the transit network and additional complexity of keeping track of each individual service route to be able to distinguish which route is disrupted (transit stops/station as nodes, transit route connections as edges with the distance/time as edge weights).

3.2 Chosen Network Construction Approach

Rather than overlaying a transit network over a road network, we'll constrain our problem as a network of transit stops/stations. This means we build our graph only from the transit dataset and do not actually use a road/track based dataset to map our network. Given the hardness of overlaying the two types of network data, we must conduct our analysis of the transit system without knowledge of the corresponding road network underneath. However, the way our transit network is constructed, all routes that pass through towards the same direction from nodes A to B are the same edge so if that edge is removed, all those routes experience a service disruption. This nuance is further discussed later in the methodology section.

We will translate the bus stops and subway stations from the transit system as the nodes of the network model and the service routes/lines that link bus stops or subway stations together will then be the edges between the associated nodes. Given this base network model, we can then introduce edge weights (discussed more in the sub-problems) that provide additional information to perform our analysis on the transit system.

3.3 Terminology and Metrics

Here we highlight the following terminology that will often be used going forward.

- The shortest paths in a weighted graph is the minimum total cost/weight to traverse from nodes A to B.
- The giant strongly connected component (GSCC) is the largest component in a directed graph in which all nodes in the component are able to reach any node in that component via a path of directed edges.
- The edge betweenness score for a particular edge is a value calculated based on the number of shortest paths that traverse through that edge in the graph.

To help us define the sub-problems of evaluating the robustness of the transit system, we briefly introduce several metrics.

- The rerouting coefficient measures how well the transit system is able to reasonably reroute the passengers from their source to their destination through the best alternative path compared to the original path.
- The transit network's strength of connectivity measures how connected the system is as a whole even after a series of disruption.
- The transit connection edge betweenness scores measure the effect on other routes after disruptions on important connections.

3.4 Sub-problems

In order to attempt this evaluation with these metrics, we need to define the following sub-problems with assumptions/constraints to reasonably simplify the computational complexity of our goal and to define a systematic approach to help build our results. The approach to these sub-problems will be discussed in the methodology section.

3.4.1 Problem 1: Simulation of disrupted service lines in transit system. For any of our metrics, we must simulate disruptions (blocks) in our network model of the transit system. A disruption may occur for a variety of reasons as mentioned above (e.g. accidents, maintenance, etc.) and could happen at stops/stations (nodes) or somewhere along the transit route connections (edges). Regardless of the reason for the disruption, we will simulate these disruptions through a selection process/algorithm of certain stops/stations (nodes) in which a disruption may occur at the node itself or between the nodes (edge).

3.4.1.1 Problem Definition. Case 1: The input of the problem is the pair of neighbouring/adjacent nodes that are selected in which the edges between those nodes are removed in order to simulate the transit disruption along that road/track. The output is the remaining network model after the edge has been removed.

Case 2: Likewise, to simulate a disruption at a stop or station, the input of the problem would be the selected node to be removed (which would necessarily remove the node's edges as well). The output is the remaining network model after the node and its associated edges are removed.

3.4.1.2 Hardness of the problem. The selection of node(s) must be handled in a methodical manner to necessarily provide meaningful results in the overall transit system. This means the process in which we select nodes must include enough sample selections of nodes to fairly represent the entire transit system. Although it is most likely possible to obtain more data about service disruptions such as historical statistics on accidents, closures for maintenance and/or traffic congestion patterns, we believe that we do not have the resources to include that additional layer in our analysis due to the constraints of time.

3.4.2 Problem 2: Measuring the rerouting coefficient for a pair of nodes from A to B. In order to obtain the rerouting coefficient, we must first define how to effectively measure the length of the passengers' trips. In this case, we use either travel distance or average travel time as our measure for travel length between the stops/stations for the weight of the edges while ideally accounting for transit transfers/connections in the travel length. It is important to consider transit transfers/connections because in a realistic context it does contribute to the transit trip's travel length. Once the measure of total trip length is defined, we will have to find the shortest path lengths (which represent the total trip time or distance) in the graph before a service disruption occurs (node and/or edge(s) removed) and in the resulting graph after the service disruption occurs.

3.4.2.1 Problem Definition. The inputs of the problem are the source and destination nodes (A & B respectively) and the data values & metric used for the weights on the edges (links) between these nodes (stops/stations) in the problem. The output is the total trip length from nodes A to B (the total sum of the edge weights in the path).

3.4.2.2 Hardness of the problem. If there is no readily available dataset provided that we can immediately use for the edge weights in our network model such as average travel time and/or travel distance between stops/stations for the transit system, we must then do some overhead work to gather the dataset (whether it is time or distance) for the edge weights. This can be an exhaustive process for a network size of N nodes (where N could be approx. 9000) with a minimum of N-1 edges (assuming a connected graph), however, there is most likely more edges existing in a transit system with N nodes given multiple connections at certain stops/stations.

While it may be trivial to measure a trip's length through the path's edge weights and algorithmically determine shortest path (single-pair shortest path problem), we would also ideally account for average transfer time (waiting time between transfers as well as travel time/distance attributed to walking between transit connections) for the total trip length to be more accurate. However, there is additional complexity in attempting to account for average transfer time at every transfer connection made given our simplified network model so this may not be implemented.

3.4.3 Problem 3: Measuring the strength of the transit network's connectivity and its resilience to targeted and random attacks. To measure a transit network's strength of connectivity, we examine the giant strongly connected component (SGCC) of the transit network each time a service disruption occurs (when an edge is removed). In order to more fairly measure this metric, we need to establish whether or not the way our network graph is constructed coincides with a realistic representation of a transit network when a service disruption occurs. This means that if an edge is removed in our network graph and disconnects a component, then it should match up with the realistic interpretation that those stop/stations (nodes) in each of those components no longer have a transit route to the other component.

3.4.3.1 Problem Definition. The input of the problem is the original graph and a list of some |E| edges to remove (determined by random or targeted by a certain graph property). The output of the problem is the value of a specified property (to be narrowed down in methodology) for each resulting SGCC with respect to |E| edge removals.

3.4.3.2 Hardness of the problem. For the output to be more reasonably realistic and meaningful, there is the additional complexity of implementing a work-around in the case that an edge removal disconnects two stops that are within very close proximity to each other that are realistically not meant to be traversed via transit with respect to each other, but by walking a short distance (a transfer). Most common case of this situation within our network is one in which there are 4 bus stops at each side of the intersection designated for a travel direction of north, east, west, south. If our graph corresponded with the road network then an intersection would be a node that encapsulates each directional stop, but this is not the case since our network was built only with a transit dataset. By default, our graph has many instances of these 4 nodes/stops within close proximity of each other. If this is not properly accounted for, it will heavily skew our results when getting the sizes of each GCC after a simulated disruption (edge removal), thus, we must implement a work-around approach in order to address this concern and mitigate its impact on our results.

3.4.4 Problem 4: Measuring how importance (Edge Betweenness Scores) changes when removing important connections in the transit network. For us to examine how edge betweenness scores change, we need to determine the edge betweenness scores for the original graph and recompute edge betweenness scores for the graphs for each edge removal that we intend to do.

3.4.4.1 Problem Definition. The input of the problem is the original transit network graph. The output of the problem are the lists of the edge betweenness scores after each previously most important edge was removed from the graph.

3.4.4.2 Hardness of the problem. Generally speaking, the computation of edge betweenness scores are expensive such that we may have to trade off reduced accuracy for reasonable computational execution time depending on size and structure of the transit network. While the number of nodes and edges of the transit graphs we examine do not go beyond 5 figures, this problem does involve us necessarily recalculating the edge betweenness scores after each removal of the most important edge. This is done to update the edge betweenness scores after edge removal and more accurately choose the most important edge to remove next. With that said, if we are doing calculating edge betweenness scores for some N iterations (number of edge removals), then we must consider a fair approximation of edge betweenness for reasonable computational execution time.

4 RELATED WORK

Wang et al [8] which further extended from Derrible's and Kennedy's study [4], conducted an analysis of the robustness of 33 metro networks across the world through several criteria based on network science and graph theory. They share a similar motivation in analyzing the weaknesses and strengths of the existing transit system (metro in their case) to determine areas of improvement through their analysis of the network properties and their theoretical metrics for their robustness criteria. In our study, we will not be basing our node/edge selection process with additional statistical data so we borrow inspiration from the attack strategies mentioned by Wang et al. using random selection or property-based selection of node(s)/edge(s) [8].

On the other hand, Abdelaty et al [1], performed a study on quantifying and classifying the robustness of 40 bus transit networks. Like the other studies mentioned above, they also used network properties such as average degree, betweenness, clustering and average path length as topological measures. Robustness measures were also borrowed and is claimed by Abdelaty to be frequently used in common literature regarding transit robustness. They classify two kinds of robustness: static and dynamic. In our case, we specifically look at dynamic robustness in which we evaluate the performance of the transit network after disruptions occur.

Zhang et al [10], conducted a study on vulnerability assessment of large-scale bus transit network (LBTN) under route service disruption which has a very similar motivation and goal as our study. One key difference between their study and ours is that while Zhang et al [10] simulate disruptions only on a singular, particular route, in our approach, the simulated disruptions affect all transit routes that go through that disrupted node or edge. In the context of our study, our simulated disruption on the transit network is an attempt to mimic a disruption on the road/metro infrastructure level where all transit routes that normally pass through a certain node/edge are affected when that node/edge is disrupted.

Ren et al [6], wrote a paper on the underestimated cost of targeted attacks on complex networks. While not necessarily written in the context of transit networks specifically, it is relevant to our study as it evaluates both random and targeted node/edge target attack strategies with respect to measuring robustness and resilience in a complex network. We leverage their research as part of our motivation and approach when we use the strength of connectivity as a metric for evaluating transit system robustness.

While the domain of above studies' research context is either just analyzing metro transit or bus transit, we go further in the sense that we look at the bus routes in tandem with the metro system as an entire network. For cities like Toronto with a very linear and simple subway system, it heavily relies on bus services/routes to supplement the limitations of the city's existing subway system. We believed it was necessary to take the bus routes and subway as a whole in our network analysis to be more fair in our evaluation since the Toronto transit system was designed with this in mind.

5 METHODOLOGY

5.1 Implementation Iterations and Steps

For measuring and obtaining the metrics needed to evaluate the network, as set throughout our requirements, the project will follow an iterative process involving data processing, analysis, and visualization.

(1) Data collection and Processing
  - Gathering of transportation network data (GTFS formats)
  - Processing GTFS into a NetworkX Graph

(2) Properties before and after attacks on the Network
  - Compute the Metrics of the network
    (a) Rerouting Coefficient
    (b) Largest Strongly Connected Component
    (c) Top 50 Edges Scored on Edge Betweenneess
  - Simulate attacks on the network based on a
    (a) Random Attack Strategy
    (b) Targeted Attack Strategy (Edge Betweenneess Scores)
  - Visualizations and Comparisons of the preceding properties / metrics

(3) Compare and Contrast with other transit Networks
  - Regather properties in Step (2) with other cities
  - Visualize and draw conclusions on the robustness of transit networks

There are many additional properties and metrics to be introduced to the problem as stated in [8], however, we will solely focus on evaluating the three properties for this report.

5.2 Challenges, Solutions, and Compromise

5.2.1 The Gathering and Processing of Transit Data: Finding datasets to build a transit network that didn't involve any manual construction was difficult to come by. Specifically, the graph required at minimum, a list of stations and a list of routes between the stations which weren't readily available.

Thus as a solution, available standardized data in the form of General Transit Feed Specification (GTFS) files were used. This contained the required information needed to build the network; however, this led to another problem where we did not have the stations that the routes are traversing through. This data was instead, overlaid as shapes that Google uses to calculate and place over their maps. To accommodate this, rather than manually going through each individual bus routes path in their pdf files (or in our case the TTC bus routes site) and Google's direction API, we used the peartree library [2] to process the GTFS [3, 7] into a NetworkX graph.

5.2.2 The Overlaying of the Transit Network onto a Road and Intersection Network: To simulate disruptions in the network, we initially wanted to overlay the transit network onto a road network. In this situation the nodes would be of two types, a bus stop or an intersection. Having such a network would allow us to generate more precise attacks, however, this was not feasible due to the data of the route stops not being readily available for the TTC as stated above.

As a compromise, we simplified the network to consist of the transit network generated through peartree where the stations are connected through links consisting of the routes average travel time.

5.2.3 Determining a Metric to use to Measure Resilience (Average Path Length vs SCC): . Initially, rather than measuring the connectivity of the transit network, we planned on measuring how disruptions will affect the overall average travel time in the network; however, this leads to some complications. As we simulate our attacks, the graph will inevitably become disconnected, preventing us from calculating the average path length over the entire graph. We thought about ignoring these edge removals, focusing on edges that won't disconnect the graph, but that prevents us from removing any targeted edges we want to remove and we wanted to avoid any restrictions on the attack strategies we used.

With that being said, this problem with connectivity spurred on the idea of measuring the size of the largest SCC. Compared to Average Path Lengths, the largest SCC measure is slightly less interesting when joined with measuring the resilience of the network. However, by measuring the size of the largest connected component we will not have any restrictions on the edges we removed. Thus, we compromised the importance of the metric for flexibility.

5.3 Network Creation and Data Collection

To build our network we required the following information from a transit system:

- Routes to calculate the paths and connections available
- Stops that represent starting points and destinations
- Stop times or distance to calculate the weight of our edges

Knowing this, we built the network using GTFS documents that provide a standard for public transportation agencies to provide to developers. This allows us to begin with one transportation network like the TTC network, and easily expand our application to compare to other transportation networks that provide a GTFS.

To build the network from a GTFS file we needed to create and map the routes to each of it's visited stations while calculating the time it takes to traverse through them. This would be reasonable for a subway system as we were able to calculate and connect the distance between stations for each line, however, with a transit network of over 9473 stops in the case of TTC [3], this is unreasonable to do manually. Thus, we utilized the following tools to process the data and analyze the network:

- Peartree to process the graph from GTFS to NetworkX
- Osmnx and Matplotlib to plot the Graph
- Networkx to analyze and manipulate the graph

Note: Peartree calculates its edge costs based on the average wait time per all stop + the trip time derived from the stop times in the GTFS between a certain snapshot in time. For our case we will use the snapshot between 7AM - 10AM.

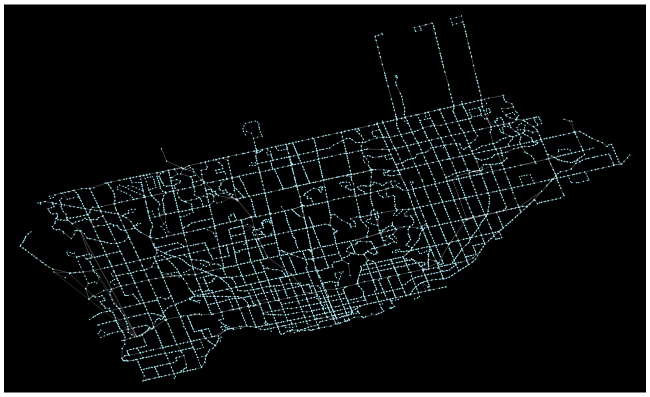

Figure 1: Toronto Transit System as a NetworkX Graph

5.4 Rerouting Coefficient

5.4.1 Algorithmic Processes. For Computing Average Travel time we iterated through each of the sampled testing stations (chosen randomly) while attacking each path directly and maintaining a list of average travel time per testing station.

(1) N = 100 source (s) and destination (d) stations are randomly sampled from graph G
(2) Compute the shortest paths in G for each of the sampled nodes using Yen's K-Shortest algorithm [9] where K=3
(3) Traverse the paths found in (2) deriving the path lengths (travel time)
(4) Find *AvgTravel(G, s, d)*, the average travel time of the K paths in G,
(5) Remove an edge from G giving us G', then recompute *AvgTravel(G', s, d)* for G' following (2) - (4),
(6) Calculate the rerouting coefficient R, from the average travel time for G and G' *R = AvgTravel(G', s, d) / AvgTravel(G, s, d)*

Note: We don't account for the cases where an edge removal disconnected the source from the destination. This is because setting the average travel time to infinity to account for unreachable stations would skew the results.

5.5 Connectivity Resilience (GSCC)

To measure the transit networks connectivity we first determined a quantitative measure to use for comparison which is the size of the largest strongly connected component (GSCC). There is much preceding research done on the resilience of a graph / network [6] that lends itself to the size of the GCC to measure a network's robustness. This leads us to applying this metric to Transit Network rather than using Average Shortest Paths which leads to some complications explained in the Section 5.2.3.

We decided to simulate disruptions in the network through the removals of connections between stations rather than removing the nodes that represent stations. This could mean a numerous amount of things; it could mean that a bus is no longer able to connect to the station because the route is too dangerous, there is too much pedestrian traffic and is no longer an option, or some for some other reasons. We find that route disruptions are more realistic and a more common occurrence compared to the unavailability of a station and is thus used to simulate attacks on the transit network.

5.5.1 Introducing Walk Edges to the Graph. It's important to note that for this specific metric our current representation of the transit network is not enough to accurately represent it's connectivity. We found that after running our simulated attacks that it disconnects drastically after removing only 0.5% of it's overall edges. Thus, here we will discuss how we addressed this issue.

In our current network we are missing important connections between stations, walk transfers. In a real world transit system people are able to walk across the road or to other stations within a certain distance. By incorporating these 'walk' edges we'll be able to create a more connected network and more importantly, a more accurate representation of the transit network.

Therefore, we introduce labels to our edges to accommodate this change and to differentiate between a transit edge and a walk edge. On top of our pre-existing transit network we add these walk edges that connect any pair of stations within a 50m radius utilizing the peartree library [2]. We assume here that 50m is the distance a person is willing to walk to a station.

5.5.2 Algorithmic Processes. In this approach we follow an iterative process, where we select a connection we want to remove either based on a random attack or a targeted attack determined by edge betweenness scores. We keep removing elements one by one while keeping track of the largest strongly connected component and repeat this process until we have either removed all the edges available or if the size of the GSCC has shrunken to 10% of the original network size.

It is important to note that we follow a similar approach to one that was mentioned by [6], more specifically we followed a variation to the edge betweenness attack strategy that they have used for undirected graphs and applied it to directed graphs.

Edge Betweenness Strategy (Given Graph G)

(1) Find the largest SCC in graph G, *largestSCC*
(2) Calculate the edge betweenness score of each edge in the Largest SCC and create a sorted ranking, *edgeRankings.*
(3) Filter out the 'walk' edges from the *edgeRankings* created in Section 5.6.1, storing them in the list *filteredEdgeRankings*
(4) Select the edge with the highest edge betweenness score in *filteredEdgeRankings* and remove it from graph G
(5) Repeat Steps 1 - 4 until the size of the largest SCC has reached 10% of the original network size

Random Attack Strategy (Given Graph G)

(1) Find the largest SCC in graph G, *largestSCC*
(2) Create an Edge list of all the edges in the graph, *edges*
(3) Filter out the 'walk' edges from the *edges* created in Section 5.6.1, storing them into *filteredEdges*
(4) Select a random edge from *filteredEdges* and remove it from graph G
(5) Repeat Steps 1 - 4 until the size of the largest SCC has reached 10% of the original network size

For this section we focused on how we simulated the attacks on the network and left out how we plotted the graph using matplotlib. For more information on how this is done please take a look at their library.

5.6 Effects of Connection Removals on Betweenness Rankings

5.6.1 Algorithmic Processes. We addressed this problem in a fairly straightforward approach and the complexity lies in visualizations rather than the algorithm. For this problem we will simply look at two snapshots in time, before and after the removal of the most important connection based on the edge betweenness scores.

(1) Calculate Top 50 most important connections based on edge betweenness scores from Graph G
(2) Remove the most important edge from G resulting in G'
(3) Calculate Top 50 most important connections based on edge betweenness scores from Graph G'
(4) Find the Difference in the Top 50 in G and G'

To visualize the results of the edge removal we used tools from peartree, matplotlib, and geopandas to highlight importance.

5.6.2 Trade-Off. For measuring this metric we traded efficiency for accuracy, in this paper we mention two types of graphs used one that includes walk cycles and one that does not. For this problem we choose to use the latter for two main reasons:

- It's a smaller graph, so we can use higher values of K, resulting in a more accurate calculation of betweenness scores
- We did not want to have the walk cycles influence the visualization of the edge importance

6 EVALUATION

To ensure that our measurements of certain properties of a transit network truly captures its robustness, we set out certain goals for each component of our analysis. These components are the data generating process, properties analysis, and data visualization; and the methods we will use to evaluate them are discussed in the following subsections.

6.1 Data Collection and Network Modeling

For the data generating process, the resulting network from our data collection and network modeling process should serve as a realistic enough representation of the transit network it is based on. To evaluate its accuracy (and relevance to a meaningful analysis), we will compare the path lengths derived from our network to travel times provided by more comprehensive mapping services, such as Google Maps. As path lengths are central to each of the properties we analyze, it is important to ensure that we measure them as accurately as possible.

We have addressed this in part by not only using the official GTFS data provided by corresponding transit agencies, but by also incorporating walk edges in our network. The inclusion of these edges allows for traversal of our model in a way that reflects the reality that people are willing to and often walk between nearby stops or stations that may be on different routes. As a result, we can better represent how "connected" the graph is in real life and include more paths with transfers than our original model would have allowed.

6.2 Properties Analysis

Given the network model, we want to be able to calculate the three main robustness properties. For any state of the network, like before and after a simulated disruption, we would like to be able to calculate the rerouting coefficient, the size of the GSCC, or the change in edge betweenness scores depending on the algorithmic process we are undertaking. After computing the values of these properties, we aim to evaluate whether the interpretations of our measurements are correct, i.e. they quantitatively describe a system's robustness in response to service disruptions.

Although we were unable to perform this component of the evaluation due to time constraints, we have outlined the steps we would take to evaluate the algorithmic processes for our property analysis in the following subsections.

6.2.1 Rerouting Coefficient Methods. To evaluate the output of the rerouting coefficient computations, we want to ensure that the result reflects how optimally the system can reroute a trip from point A to point B whenever possible. As the node selection process is random, it is suitable for producing an overall result that reflects the entire system. To verify whether our calculation of the rerouting coefficient is correct, we will consider the evaluation of computed shortest path lengths and average shortest path lengths. As mentioned previously, we can use more comprehensive mapping services to evaluate the relative accuracy of the shortest path lengths. As obtaining the shortest path length after disruptions may require a time consuming workaround of these systems, it is only feasible to compare a small but representative set of (s, d) nodes. As for the average shortest path lengths, it is important the calculations of the ratio of averages for the original and modified graph using the same number k top path; otherwise, the results may be skewed too low.

6.2.2 *Connectivity Resilience Methods.*

As mentioned in the previous sections, our approach to understand the resilience of the network has been adopted mostly from [6]. As we mention in Section 5.2.3, this decision stemmed from the fact that our attack strategies would disconnect our graph and skew our results for average travel time, the original metric we proposed for measuring robustness. While the research in [6] shows that connectivity is an appropriate measure for resilience, we like to further evaluate and analyze whether our algorithm's results perform as an indicator of the trends in average travel time as the graph becomes more disconnected. To do this, we plan to use some Erdős-Rényi graphs as null models that have the same number of nodes and edges as some of our transit networks. With these null models, we will perform the same random and targeted attack strategies done by [6]. At each iteration of the strategy, we will not only measure the size of the GSCC but also its average travel time for each strongly connected component of the graphs. After this, we plan to plot our results for size and average travel time. After looking at our results, we will attempt to verify whether a decrease in graph connectivity indicates a proportionate decrease in average travel time.

6.2.3 *Edge Betweenness Rankings Methods.*

To evaluate whether the tradeoff for efficiency for accuracy is appropriate for the result of our algorithmic process, we will try to evaluate how close the approximation of the edge betweenness scores are for each iteration. To be more specific, in some of iterations that we calculate the edge betweenness scores, we will also calculate the edge betweenness again using a higher value of k for the top 5 ranked edges. We do this because even the top 5 should be sufficient but also important for the analysis of the most important edges. As the top 5 represents 10 % of the ranked edges, we feel it is safe to assume if the margin of error between our less and more accurate approximations is small, then our tradeoff of our method is practical not only for the top 5 but also for the top 50 that we rank. Again, because we only do this for some iterations (perhaps chosen randomly to generally represent all of them), this type of evaluation is computationally inexpensive compared to calculating scores using all nodes in the graph for every iteration.

6.3 Visualization Process

The process of visualizing our results should produce graphics that highlight the general trends in network properties in response to disruptions and should also allow us to identify areas of structural improvement for a transit network that we are interested in. We aim for this process to be generalizable between transit systems so that we can visually compare the robustness of ones in this analysis and perhaps others considered in future works.

We mention again that we have been able to do as such using tools like as peartree,osmnx, and geopandas to visualize the modeled network as well as matplotlib for plotting the results of our properties analysis. Indeed, these results are included and compared in the following conclusion section.

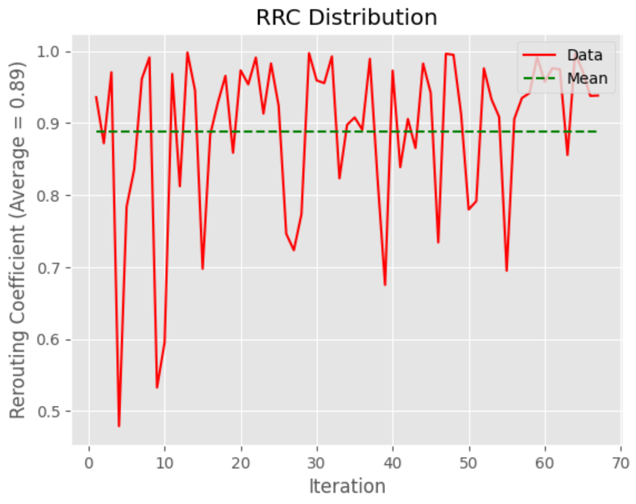

Figure 2: Toronto Transit Commission (TTC) Rerouting Coefficient Distribution (n=100)

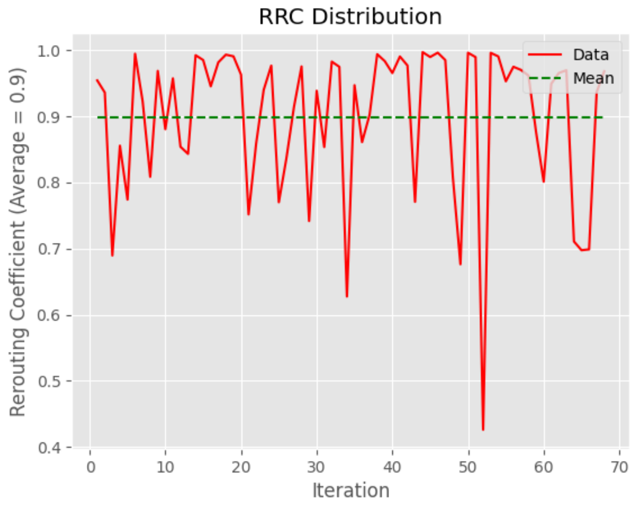

Figure 3: Réseau de transport de la Capitale (RTC) Rerouting Coefficient Distribution (n=100)

7 CONCLUSIONS

7.1 Analysis of Preliminary Results

Upon inspection of the TTC and RTC travel time distribution in Figures 2 and 3, we see that they have a similar Rerouting Coefficient. Given a random simulated disruption on a bus route, the average travel time increased by 10 percent in both Toronto's and Quebec's transit system. However, we found that these attacks on the network caused more major disruptions in Quebec's transit system. This led us to do further network analysis on the two systems to better understand their robustness. We'll need to evaluate more transit networks to compare and run more iterations of our Rerouting Coefficient algorithm to determine the efficacy of our

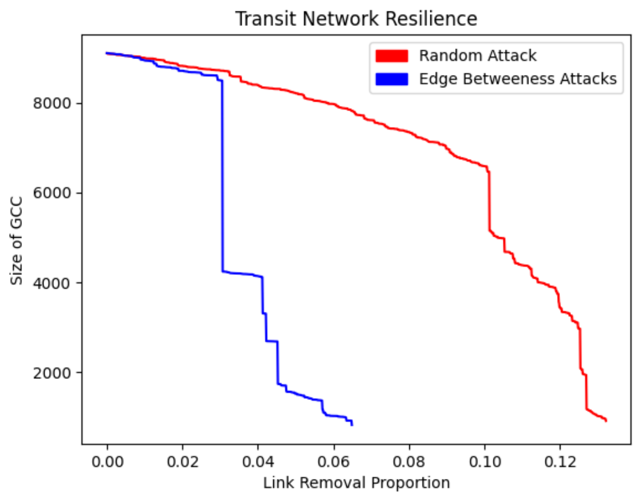

Figure 4: Toronto Transit Commission (TTC) GCC Resilience to Edge Removal Distribution

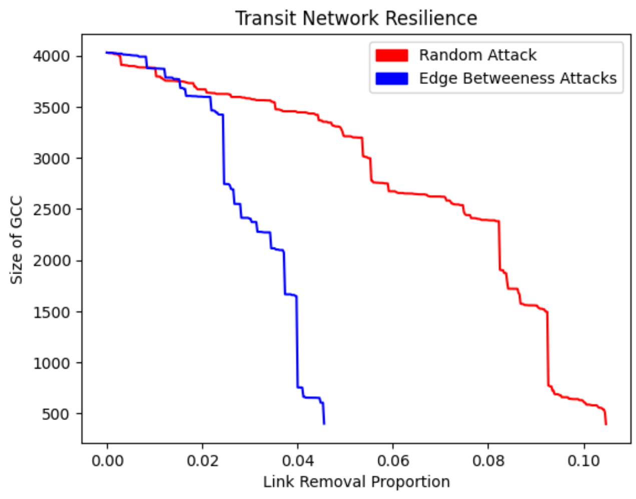

Figure 5: Réseau de transport de la Capitale (RTC) GCC Resilience to Edge Removal Distribution

Rerouting Coefficient in terms of evaluating the robustness of a transit network.

Looking at the strength of connectivity of the transit networks, we took our initial network and improved the accuracy of the representation of the transit network by introducing walk edges. After simulating random attacks, as well as edge betweenness based attacks, we found that the TTC was more robust in terms of its ability to maintain a strongly connected component by about 2 percent compared to Quebec's transit system (Fig 4 , Fig 5).

Something interesting to note is when we removed the edge with the highest edge betweenness, 14 out of 50 of the top 50 edges (with respect to edge betweenness scores) were different. Initially, the most important edge was located on a major road in Toronto, Yonge

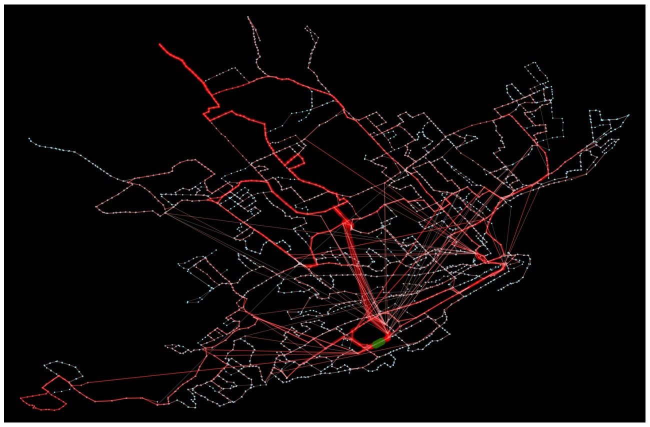

Figure 6: Réseau de transport de la Capitale (RTC) Edge Betweenness Distribution

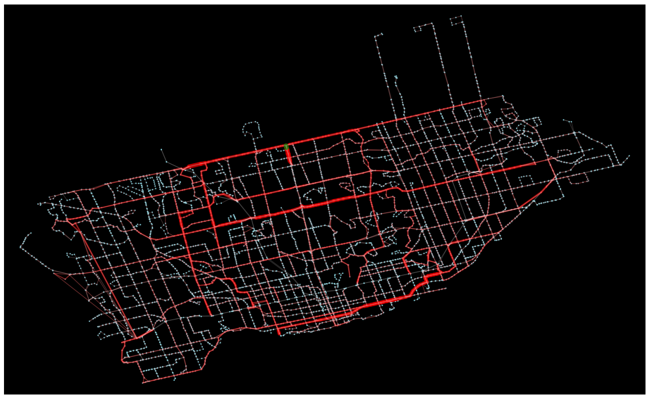

Figure 7: Toronto Transit Commission (TTC) Edge Betweenness Distribution

St. However, after the edge removal, the most important edge was located on another road, York Mills. As an extension of this analysis on the network's edge betweenness, we hope to analyze historical data on disruptions that have occurred on these important edges, and the impact on traffic during that particular instance.

7.2 Future Works

For future works, we plan to include community detection as a metric to measure the efficiency of the network. We'd like to analyze and evaluate strong sectors in the transit system and find the links that are connecting them with the use of edge betweenness. These properties will provide us with valuable information that could be used to improve the transit networks. For example, with communities and edge betweenness we would be able to:

- Find highly traversed routes (Can increase frequency of these routes)
- Find well covered and weakly connected communities (Can develop more connections to improve small communities)

We'd also like to expand our attacks on the network and overcome some of the naive assumptions of the network. Rather than attacking only once per traversal, we'll work on taking multiple attacks and evaluate how the travel time reacts to these random and targeted attacks. We'll either do one of the following:

- Similar to what we have done, have one traversal pertaining multiple paths and start removing edges either targeted or random or;
- A more expensive method, calculate the average travel time of the entire network in G, then calculate the average travel time in G - 1 edge, average travel time for G - 2 edges, ... and the average travel time for G - n edges

After plotting the average travel times for the different cities, we'll be able to find trends and compare the different networks. Additionally, a possible extension to this project is to look at how global transit agencies deal with this type of information network. For example, using our network data in combination with historical ridership data, how have different transit agencies managed severe bottlenecks in their transit networks (transit expansions, alternative routing, alternative modes of transportation). Implementing transfer times to our generated networks would give a more realistic view of the transit system, since transfer times can have a noticeable effect to the average travel time of passengers [5]. Furthermore, our aim is to continually find impactful metrics with regards to the transit system and find ways to evaluate them.

REFERENCES

[1] Hatem Abdelaty, Moataz Mohamed, Mohamed Ezzeldin, and Wael El-Dakhakhni. 2020. Quantifying and Classifying the Robustness of Bus Transit Networks. *Transportmetrica A: Transport Sciences* 16 (02 2020), 1176–1216. https://doi.org/10.1080/23249935.2020.1720042

[2] Kuan Butts. 2018. Peartree. https://github.com/kuanb/peartree.

[3] Toronto Transit Commission. 2020. *TTC GTFS.* Retrieved Oct 6, 2020 from https://open.toronto.ca/dataset/ttc-routes-and-schedules/

[4] Sybil Derrible and Christopher Kennedy. 2010. The complexity and robustness of metro networks. *Physica A: Statistical Mechanics and its Applications* 389 (09 2010), 3678–3691. https://doi.org/10.1016/j.physa.2010.04.008

[5] Changhee Kim, Soo Wook Kim, Hee Jay Kang, and Seung-Min Song. 2017. What Makes Urban Transportation Efficient? Evidence from Subway Transfer Stations in Korea. *Sustainability* 9, 11 (2017), 2054. https://doi.org/10.1007/978-3-642-96868-6_25

[6] Xiao-Long Ren, Niels Gleinig, Dijana Tolić, and Nino Antulov-Fantulin. 2018. Underestimated Cost of Targeted Attacks on Complex Networks. *Complexity* 2018 (2018), 1–15. https://doi.org/10.1155/2018/9826243

[7] transitfeeds. 2020. *RTC GTFS.* Retrieved Oct 6, 2020 from https://transitfeeds.com/p/reseau-de-transport-de-la-capitale/40/latest/

[8] Xiangrong Wang, Yakup Koç, Sybil Derrible, Sk Nasir Ahmad, Willem Pino, and Robert Kooij. 2017. Multi-criteria robustness analysis of metro networks. *Physica A: Statistical Mechanics and its Applications* 474 (05 2017), 19–31. https://doi.org/10.1007/978-3-642-96868-6_25

[9] Jin Y. Yen. 1971. Finding the K Shortest Loopless Paths in a Network. *Management Science* 17, 11 (1971), 712–716. http://www.jstor.org/stable/2629312

[10] Kin Zhang, Huiying Wen, Jian Lu, Shubin Li, and Da Lei. 2020. Vulnerability assessment and visualization of large-scale bus transit network under route service disruption. *Transportation Research Part D: Transport and Environment* 88 (2020). https://doi.org/10.1016/j.trd.2020.102570