Urban Spiderwebs: Detecting and Modeling Traffic Congestion within Urban Road Networks

Owen Wang, Emily Ling

CS 224W Final project, December 2016

1 Introduction

As city populations continue to swell, city planners and civil engineers need to match the tide of incoming people with scalable transportation systems that can support a growing population, or else face gridlock congestion and immobility undesirable for inhabitants and visitors.

In our project, we will use measures of graph centrality such as betweenness and closeness centrality to identify which areas of a city's road network has the greatest potential for traffic congestion for vehicles. Furthermore, we will create traffic simulations for dynamic, empirical insight and visualizations into which streets experience the most vehicle traffic. After drawing conclusions from our results, we will compare and contrast the different approaches we used to measure traffic congestion, as well as the differences we observe in cities across the world and their points of peak traffic.

2 Related Works and Research

2.1 A Google-like Model of Road Network Dynamics and Its Application to Regulation and Control

Crisostomi, Kirkland, and Shorten's work focuses on modeling road networks as Markov chains, both for the purposes of traffic simulation and network congestion detection. Using a *dualrepresentation*, they represent roads and streets as nodes, and intersections as a web of edges among the streets that meet at the junction. This formulation allows for information about turning probabilities (data would have to be provided), which turns are legal, and street metadata to be encoded easily in the graph. The time T (in time steps) one takes to traverse a street was simulated by a self-edge at each node; with probability (T-1)/T a traveller would take the self edge, otherwise it will take one of the edges turning onto a different street. By turning the road network into a matrix of state transition probabilities, the authors observe the Perron vector represents the long-term stationary distribution for each state in the network.

The authors use a dynamic strategy to calculate road congestion reminiscent of PageRank role in scoring websites, and different from the static measures of centrality most commonly discussed. We drew inspiration from Crisostomi, Kirkland, and Shorten's work to create our own traffic simulation, and were especially captured by their idea of using a dual representation of a road network to run simulations. However, where they analyze the long-term stationary vector of the road network as a matrix to diagnose congestion, we proceeded with a more empirical method to discover areas of the graph with high traffic. For more information, see "6.2 Modeling Traffic Units".

2.2 Access to destinations: travel time estimation on arterials

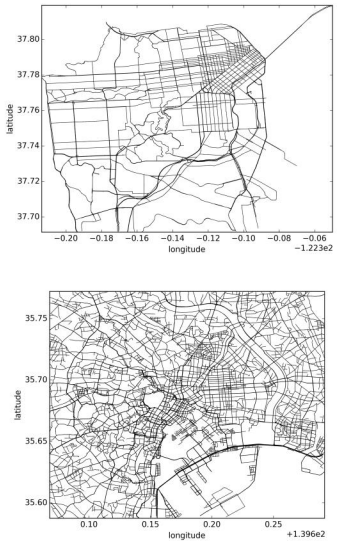
Again for modeling traffic, an important paper is Xiong's and Davis' 2007 work. The authors' paper focused on identifying and evaluating several different models for estimating travel times on *arterials*, or signaled roads with traffic lights. Travel time depends on factors like road capacity, the current volume of traffic, and free-flow speed without traffic. Some models included the Bureau of Public Roads (BPR) function,

the Singapore model, the conical volume-delay model, and the Skabardonis-Dowling model. The authors found that the Singapore and Skabardonis-Dowling models were two of the models that performed best in predicting travel time when tested against field data.

Xiong and Davis provide several promising traffic functions that have been used in the past in various contexts to estimate travel time delay due to congestion. In our traffic model, we use a nonlinearly decreasing function for traffic congestion delays similar to both the BPR function and the Skabardonis-Dowling model. We kept in mind that Xiong's and Davis' work is on arterial links however, and their results have not been tested on traffic patterns in freeways and roads with fewer stop signals. Hopefully, our traffic function will prove effective for both arterial roads and highways both.

3 Obtaining Road Network Data

Figure 1: Road Network of San Francisco, California and Tokyo, Japan



In order to obtain data on urban road networks in many different countries and regions of the world, we use OpenStreetMap (OSM), a crowd-sourced project that collects extensive geographic data about roads, trails, buildings, and more, from all over the world. OpenStreetMap is built by a community of contributors who use field maps, GPS devices, and aerial imagery to maintain accurate information, and is distributed freely under the Open Data Commons Open Database License. The data used for this project was obtained from Mapzen's Metro Extracts, a collection of the most recent OpenStreetMap data for 200 popular cities that is updated on a regular basis. OpenStreetMap data comes in an OSM XML file format, and contains a list of nodes, a list of ways, and a list of relations. *Nodes* are a single point in space defined by latitude and longitude; nodes form *ways*, which are an ordered list of nodes that define features such as roads, railways, and areas. Nodes and ways each have tags that describe characteristics of the node or way such as the name, the type of road, and more.

For the purposes of this project, we picked 25 different cities from many different regions of the world. The following is the list of cities:

```
Asia: Shanghai, Beijing, Tokyo, Seoul, Jakarta
Europe: London, Berlin, Saint Petersburg, Rome, Amsterdam
North America: New York, San Francisco, Los Angeles, Vancouver, Mexico City
South America: Rio de Janeiro, Sao Paulo, Santiago, Bogota, Buenos Aires
Africa: Nairobi, Cape Town, Cairo, Addis-Abeba, Accra
Middle East: Istanbul, Doha, Jerusalem
Oceania: Sydney, Auckland
```

In choosing these cities, we strove to choose fairly large or well-known cities representing each region of the world, with some diversity in terms of population, size, and history. The purpose was to pick cities that were fairly representative of the region, in order to be able to compare and contrast cities in different regions.

For the rest of this paper, the reader should keep in mind that references to a *street* may not be a street in the common sense of the word: in our road networks, one real-life street may be subdivided into several street segments to replicate the curves and twists on a road, for example. This is directly because one way in the OSM data has a sequence of nodes outlining the curve of the street.

We use the nodes and ways to build a SNAP graph. The graph only considers the ways tagged as a "highway", which include roads of all types, and ignores other ways such as buildings, railroads, and barriers. Each SNAP node corresponds to an OpenStreetMap node and represents a street intersection. Since the nodes making up a way are ordered, there is a SNAP edge between each pair of two nodes forming the way, representing a street between the two nodes. The road networks were visualized using matplotlib's plotting functionality, where each edge is represented by a line on the map (Figure 1).

4 Measures of Traffic Congestion

We compared our selected cities across a spectrum of different measures, chosen to give insight into where high-traffic areas around the city area.

4.1 Betweenness Centrality

The betweenness centrality of a node is the number of shortest paths in the graph (from every node to every other node) that traverse that node. We leveraged the SNAP library's built-in snap.GetBetweennessCentr() function for this purpose.

The betweenness centrality captures how "well-trodden" a path or intersection is, as it is proportional to the number of pairs of nodes that must travel over that path or intersection to arrive at each other. We assume most travellers in the network will use the shortest path they can to arrive at their destination, and thus an edge or node with high betweenness centrality is suggestive of high volumes of traffic in that area. Inversely, those with low betweenness centrality are likely to indicate low traffic. We expect regions near major, centrally-located roads (such as freeways and their entrances/exits) to rank high on betweenness centrality.

4.1.1 Modifying Betweenness: Approximate Weighted Betweenness Centrality

In addition to using the traditional betweenness centrality in road networks, we created our own variant of the measure to circumvent two issues.

With road networks, there is a major challenge to incorporate street weights into calculations of closeness, betweenness, and the like. Whereas in another graph (e.g. social networks) edges are fundamentally equal, an edge in a road network can be very short or very long in distance, and this crucial information is lost.

A second challenge we experienced was long running times. The betweenness centrality algorithm requires finding the shortest paths from every node to every other node, a non-trivial operation.

We resolve the first problem by incorporating distance into calculations of shortest path between nodes during the betweenness centrality algorithm. We resolve the second by sampling at random 1/100 of all the pairs of nodes in the city graph. The rest of the betweenness algorithm remains the same as before. Our *approximate weighted betweenness centrality* leveraged sample code from Question 2 of Problem Set 4 in this class.

4.2 Closeness Centrality

The closeness centrality of a node N is the average length of the shortest path between N and every other node in the network. A lower closeness centrality indicates a node occupies a more central location in the graph, with fewer extremely long shortest paths to another node. To calculate the closeness centrality of nodes, we leveraged SNAP's snap.GetClosenessCentr() function.

We expect nodes that are geographically closer to the center of a city to possess a higher closeness centrality. Nodes with a high closeness do not necessarily have a high betweenness; there may be pockets of nodes near a high betweenness way in the network that receive little to no traffic. Indeed, we expect that nodes and edges with high betweenness are closer to the center of cities, but that only a subset of nodes with high closeness centrality in the center of cities will also have high betweenness. The nodes closest to those with high betweenness may in fact experience very low betweenness themselves, in the way that the smaller, local roads next to major highways have less traffic because of their adjacency to them.

4.2.1 Modified Measure: Approximate Weighted Closeness Centrality

We similarly used a variant of closeness centrality in addition to the original measure for the same reasons we did for betweenness centrality (see section 4.1.1).

4.3 New Measure: Urbanness

The urbanness measure of a node N in a graph is defined as the reciprocal of the average distance from N to its nearest 500 nodes, as traversed in a weighted Dijkstra search from N. Here we define the distance between u and v not as the graph distance (i.e. number of edges traversed to travel from u to v), but as the physical distance between the two given their associated latitude and longitude coordinates. More formally, we define urbanness as follows:

$$urbanness(N, K) = \frac{K}{\sum_{v \in D(N,K)} distance(N, v)} \tag{1}$$

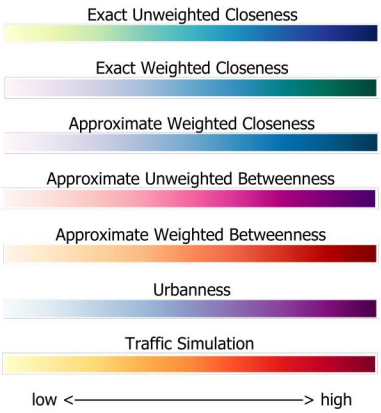
K here is the number of nodes to traverse outwards from N; a small K fails to capture nearby parallel streets or densely packed blocks, but a large K creates an unreasonably long runtime.After tuning the parameters a bit, K=500 was found to give a good balance between detecting dense urban areas and reasonable runtime. $D(N, K)$ here represents the set of the first K nodes encountered in a Dijkstra search starting from the node N. The weight of an edge is defined as the physical distance between two points. Since the relative curvature of the Earth's surface is almost negligibly small in a geographic region the size of a city, we simply use the Euclidean distance between the latitude and longitude coordinates of two nodes in R2, as defined below:

$$distance(u, v) = \sqrt{(u_{latitude} - v_{latitude})^2 + (u_{longitude} - v_{longitude})^2} \tag{2}$$

Intuitively, the urbanness of a given node was designed to capture how concentrated the nodes near it are. Many nodes clustered in one region can be indicative of a high concentration of intersections and therefore traffic congestion. We expect nodes with high urbanness to appear around downtown areas and compact residential neighborhoods.

5 Traffic Simulation

Figure 2: Legend



The next stage of our project was to generate data from traffic simulations in the cities we had analyzed. Static graph measures such as betweenness and closeness had already revealed insight into the topography of a city and its most travelled, most congested areas. Next, we proceeded with our own traffic simulation model. Not only does this traffic simulation model provide a way to simulate real traffic data in city networks for the purposes of hypothesis testing and comparison to static measures, it itself may prove to be a reliable predictor of traffic congestion.

5.1 The Dual Graph

Inspired by the work of Crisostomi et al. in our readings of related research, we performed the traffic simulation on a dual graph representation of a city's traditional road graph, where the usual representation of nodes and edges are reversed: all streets are represented as nodes instead of edges, and an intersection of two or more streets is represented as a complete graph between the street nodes that meet at the intersection. (From this point on, the reader can assume references to a city's or simulator's graph refers to a dual graph representation of the city's road network.) This choice allows for several advantages.

We make the assumption that vehicles spend most of their time when driving travelling down streets, not stalled or turning an intersection. To reflect this, a car's state will be tracked by the street it is traversing at any time instead of an intersection. The dual graph representation allows us to easily do this by making streets into nodes, and also associate with a street node its length, geographical coordinates, and other useful information. Additionally, it's possible to encode further information such as turning probabilities and illegal turns into edge weights and the absence of edges at an intersection. Our OpenStreetMap data unfortunately did not include data for us to leverage the latter in this case, but the dual graph representation was a good choice for the former nonetheless.

The dual graph class was created through SNAP. We begin with a empty snap graph D, and let R be a traditional road graph we parsed from OSM. For every edge e in R, we assign it a new node ID $p(e)$ unique from any other ID before it. This association is remembered, then $p(e)$ is inserted into D as a node. Information about the street's coordinates (averaging endpoints) and its weight (length) is associated and recorded with the new node in D.

Then for every node n in R, we identify the edges $E(n)$ that meet at n by iterating over n's neighbors. Let $E'(n)$ be the set of node IDs $p(e)$ associated with every e in $E(n)$. We create edges between every pair of nodes in $E'(n)$ and insert them into D.

Figure 3: San Francisco Traffic Congestion Measures
(left column): Unweighted closeness, exact weighted closeness, exact unweighted closeness.
(right column): Unweighted betweenness, weighted betweenness, urbanness

5.2 Modeling Traffic Units

To complete our traffic simulation model, we created two other classes: one for a car that will move along our dual graph, and one for a traffic simulator to coordinate the entire operation with the traffic units and dual graph.

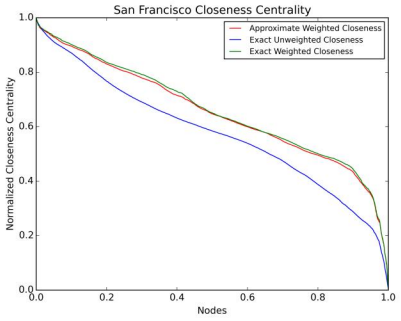
5.2.1 The Car class

Upon initialization, a car is connected to its parent simulator, from which it can access the city's graph. The car is then initialized with a randomly generated journey in the city. Two streets with lengths under the median street length in the city are selected at random. We do this to prevent cars from selecting a large highway or major road in the city as a source or destination; the assumption is that most cars will start and end at smaller streets like a home, school, park, or store. Using the median street length as a threshold allows both a minimally limited selection pool and a sufficiently low absolute threshold. Then, the car is given its *itinerary*, a list of streets to traverse to arrive at its

destination starting from its source; it is properly ordered. The itinerary is discovered by A* search with a heuristic function of the euclidean norm distance (in geographical coordinates) to the destination.

The car keeps track of which street in the itinerary it is currently on with an index, as well as its *progress* (a non-negative float) down the current street. When signaled to advance one time step, the car will increment its progress by some amount (see below), and when the car's *progress* quantity exceeds the length of the current street it will move onto the next street in the itinerary. If the next street is the destination, the car's journey is complete. At this point, if the simulation is not yet over and there are still time steps remaining, the car is "reincarnated" by being assigned a new starting point, ending point, and itinerary.

Figure 4: San Francisco Closeness Centrality



The value of *progress*' increment inside each car is intuitively a measure of speed, how fast it can traverse streets. *increment* can be summarized as a value randomly from 1/8 to 1/12 of the current street's length and potentially scaled down by a factor for traffic, but is at minimum 5 and at maximum 15:

$$t = 0.25^{(count-1)/L}$$
$$i = t * L/(10.0 \pm 2r)$$
$$increment = min(15, max(5, i))$$

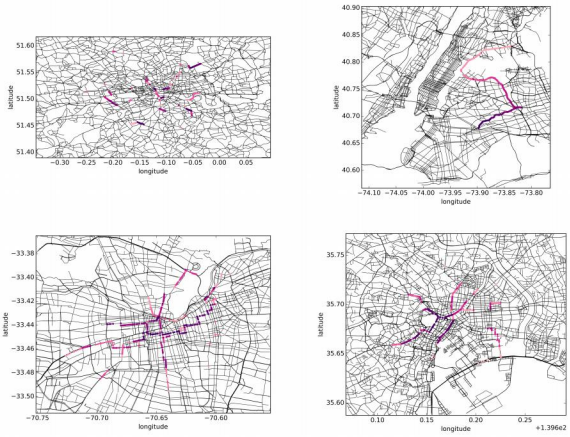
where L is the length of the car's current street, r is a random uniform variable from 0 to 1, and *count* is the number of cars (including our current car) on the street. t is a traffic coefficient the car requests from the parent simulator at each time step: an exponentially decreasing function starting at 1.0 when no other cars are on the current street, and 0.25 when there are as many cars on the current street as the street's unitless length.

5.2.2 The TrafficSimulator class

The traffic simulator is initialized with a graph of a city G, a number of cars N, a number of time steps to run the simulation for T, and a frame rate F for data collection. It also contains a dictionary of street node IDs to car counts for streets in G with at least one car for traffic coefficient calculation, and data structures for collecting data to vizualize our traffic simulation with (see next section).

The traffic simulator initializes a list of N cars, with itinerary and all, described in the last subsection. Then for T time steps, at each time step it signals all N cars to increment forward on their journey (and reincarnate as mentioned before, if necessary). As each car moves through its itinerary, it diligently updates the simulator's car counts dictionary to maintain accurate counts when it moves from one street to another.

Figure 5: Unweighted betweenness centrality (clockwise from upper left) London, England; New York, New York; Tokyo, Japan; Santiago, Chile

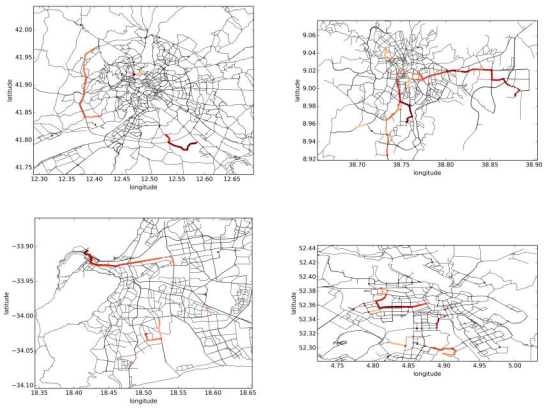


5.3 Collecting Data on Simulated Traffic

The traffic simulator is in charge of taking snapshots of simulated traffic while the simulation is running. From its frame rate F, the simulator will take a snapshot of the current cars' positions every F time steps. A *snapshot* of a running simulation at some time step yields a Counter of non-empty streets' (>= 1 car) coordinates to the count of cars on that street.

We collected traffic data in two ways: an overview of long-term traffic distribution in the city, and a series of snapshots that reveal the flow of cars in the city over the duration of the simulation. For long-term traffic data, we aggregated all snapshots the simulator took into one dictionary storing overall car counts over the run of the simulation and create a plot from it.

Figure 6: Weighted betweenness centrality (clockwise from upper left) Rome, Italy; Addis Abeba, Ethiopia; Amsterdam, Netherlands; Cape Town, South Africa



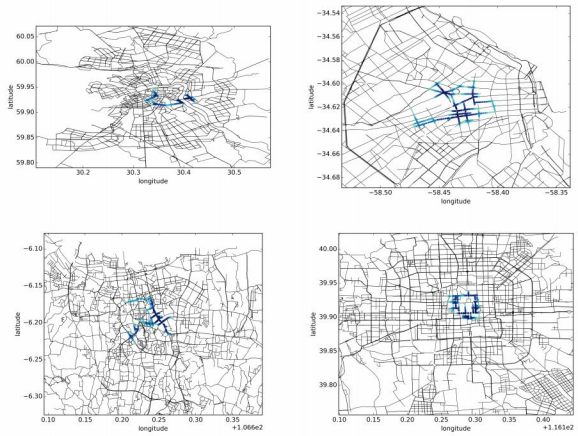
The second way we collected data, the sequence of separate snapshots, was used to create chronological visual demonstrations of cars flowing through a city. Taking every frame 10 time steps apart, one can clearly see the movement of cars on a map when plotted. We then created videos from these series of plot images to easily display the movement of the cars.

6 Results and Conclusions

6.1 Traffic Congestion Measures

For each traffic congestion measure, we plotted on a map the top 500 nodes with the highest values for that measure. The nodes were colored according to the legend of gradients in Figure 2.

Figure 7: Unweighted closeness centrality (clockwise from upper left) Saint Petersburg, Russia; Buenos Aires, Argentina; Beijing, China; Jakarta, Indonesia

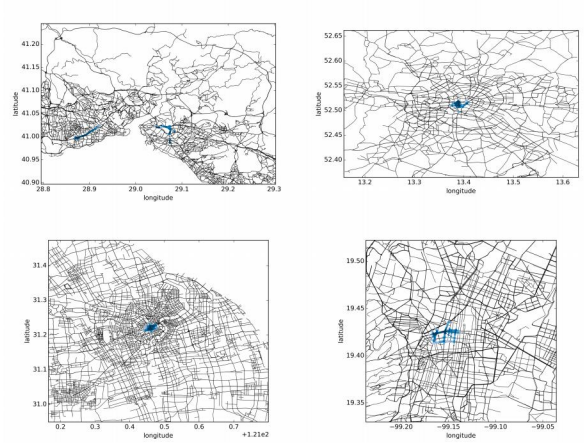


6.1.1 Applying Results to San Francisco

We begin by first examining the results of the congestion measures based on a nearby city, San Francisco. Figure 3 displays the six traffic congestion measures overlaid on a map of the city. By doing so, we can gain more insight into the results by matching the highlighted nodes with actual geographic areas. Firstly, the closeness measures did fairly well at identifying important downtown areas, as we predicted in the methods section above. The unweighted closeness measure focused on Fillmore District, Mission District, and City Hall/Civic Plaza area, which are fairly popular, lively areas of San Francisco. The two weighted closeness measures identified a slightly different region, the Castro District. The approximate closeness measure performed nearly identically to the exact measure with a significantly faster run-time, indicating that sampling just 1% of nodes in a road network is more than sufficient to give highly accurate estimate of closeness. This is further demonstrated in Figure 4, which sorts the normalized closeness for all nodes in the graph and plots these values. Note that the lines for exact and

approximate weighted closeness are nearly identical, while unweighted closeness has a smoother distribution with a less steep drop-off in closeness at the end.

Figure 8: Approximate weighted closeness centrality
(clockwise from upper left) Istanbul, Turkey; Berlin, Germany; Mexico City, Mexico; Shanghai, China

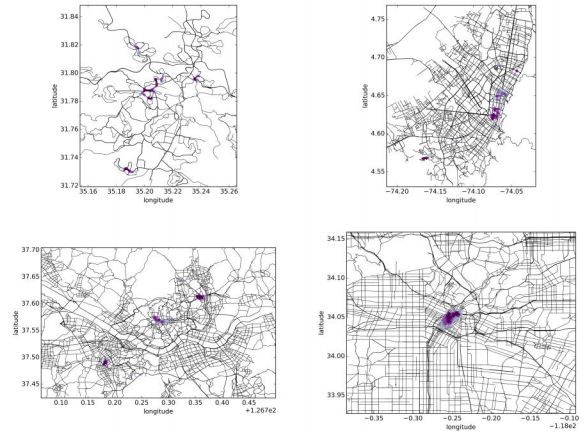


Returning to Figure 3, the betweenness measures were effective at identifying major, important throughways for the city. The unweighted betweenness identifies roads such as CA-1, Divisadero Street, Guerrerro Street in the Mission, part of highway 101, and Mission Street. The weighted betweenness identifies also identifies Guerrerro Street in the Mission and Mission Street, but focus more heavily on Lincoln Way and Highway 280. These roads are fairly important roads that help to link different parts of San Francisco together. It is interesting that both the weighted and unweighted measures tended to highlight smaller roads rather than highways; one explanation for this is that highways are generally represented in OpenStreetMap data as multiple parallel roads, so each parallel segment shares the total betweenness, causing it to be lower than expected. Additionally, the graphs were undirected, while cities frequently have many one-way roads, so the shortest paths in our graph include a larger than expected number of small one-way roads rather than highways. Nonetheless, the roads identified do serve as important roadways for San Francisco.

The last graph in Figure 3 displays urbanness in San Francisco. At a first glance, two major sections are highlighted: the Financial District and the South of Market region near City Hall. These areas are also among the most densely packed regions of the map simply from a visual inspection, and based on background knowledge of San Francisco, these regions are indeed highly urban and dense areas. Thus, the urbanness measure seems to effectively identify downtown regions of a city, which may indicate higher levels of activity or congestion.

6.1.2 Comparative Analysis

Figure 9: Urbanness
(clockwise from upper left) Jerusalem, Israel; Bogota, Colombia; Los Angeles, California; Seoul, South Korea



For each measure, we present the maps of 4 selected cities. Figure 5 displays the unweighted betweenness centrality for London, New York, Tokyo, and Santiago. These four cities, which represent four different continents, each demonstrate different patterns of betweenness. New York, like many North American cities, had highly concentrated betweenness along a single or a very small number of major roads. This suggests that New York and similar cities may have large amounts of traffic along a small number of major roads. South American, African, and Asian cities generally had concentrated betweenness along several roads emanating out from the city center, such as in Santiago or Tokyo. This suggests that these cities may have significant traffic moving into and out of the city center.

On the other hand, London as well as several other European cities have very scattered pockets of concentrated betweenness, suggesting that there are not particularly large, major roads, so traffic may be more evenly dispersed. Figure 6, which maps weighted betweenness centrality for Rome, Addis Abeba, Amsterdam, and Cape Town, displays some of these traits as well.

Figure 7 displays unweighted closeness centrality, and Figure 8 displays approximate weighted closeness centrality, which serves as an equivalent substitute for exact weighted closeness, as described in the section

above. In general, closeness typically seems to identify one central downtown area of the city. The size and shape of the downtown area varies; for example, Beijing displays a unique ring-structure that causes the closeness centrality to be concentrated around a central circle. Unweighted closeness centrality generally behaves fairly similarly to weighted closeness but locates smaller and more concentrated regions of high centrality. An interesting outlier is Istanbul in Figure 8, which has two highly disparate regions of high closeness that are separated by a body of water.

Figure 10: Closeness Centrality Distribution

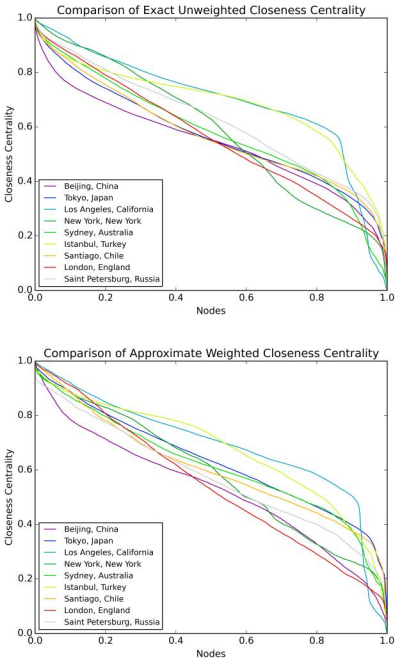
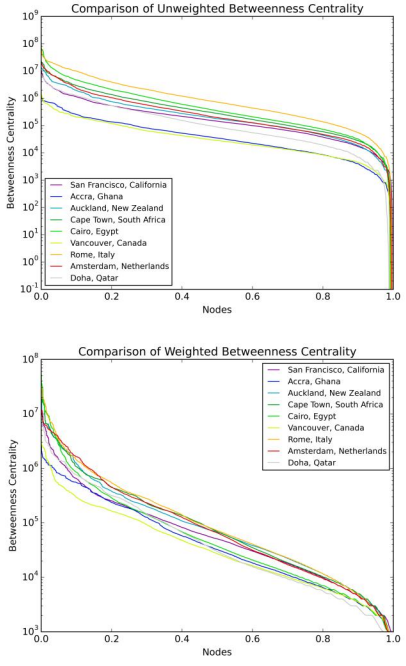


Figure 9 maps urbanness for Jerusalem, Bogota, Los Angeles, and Seoul. In some cities, such as Los Angeles, there is a single clear "downtown" area that tends to have both high urbanness and high closeness centrality. Bogota, on the other hand, like Buenos Aires and other South American cities, has 2-3 smaller areas of high urbanness that are a bit different from the results from closeness centrality. Finally, cities such as Seoul, Jerusalem, and other European and Asian cities tend to be generally more evenly dense, and thus have several small areas of urbanness that are scattered around the city.

Figures 10 and 11 display distributions of weighted and unweighted betweenness and closeness centralities in a varied selection of cities. From these graphs, we observe that most cities share very similar distributions for both betweenness and closeness, with Los Angeles being a slight outlier in closeness. We also see that weighted betweenness seems to be distributed more evenly with a less sharp drop-off at the end. Weighted and unweighted closeness centralities appear to be fairly similar in distribution.

Figure 11: Betweenness Centrality Distribution



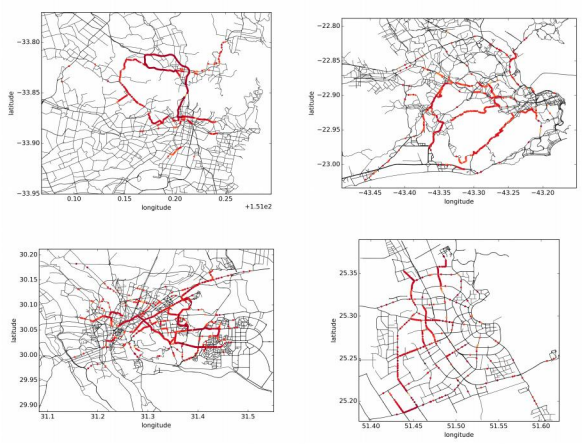
6.2 Traffic Simulation Results

Figure 12 displays the results of traffic simulation data for four cities. Simulations were run with 10,000 cars in the city for 10,000 generations, taking a snapshot every 10 time steps. The 1000 snapshots' data were overlayed and combined into one map of streets

representing their relative long-term traffic, yielding a Counter of 10,000,000 car counts distributed over streets in the city from all snapshots. Streets that experienced more traffic over the generations will have higher car counts and a darker red color; those with less traffic a lighter red color. In the maps for Sydney and Rio de Janeiro, there is high traffic on streets that cross bodies of water, which makes sense as all traffic traveling between the two areas of land must be routed through one of a small number of roads, leading to traffic congestion. Other cities have slightly more distributed traffic, such as Cairo, indicating that traffic is spread out over different roads and thus is less concentrated

7 Works Cited

Figure 12: Traffic
(clockwise from upper left) Sydney, Australia; Rio de Janeiro, Brazil; Doha, Qatar; Cairo, Egypt



Crisostomi, E., S. Kirkland, and R. Shorten. "A Google-like Model of Road Network Dynamics and Its Application to Regulation and Control." International Journal of Control 00.00 (2010): 1–26. Print.

Davis, Gary A., and Hui Xiong. "Access to Destinations: Travel Time Estimation on Arterials - ITS Research Reports." N.p., 2007. Web. 18 Nov. 2016.

8 Team Contribution

Team contribution:
Emily: collecting and downloading data, parsing OSM files, building SNAP graphs from OSM, all traffic congestion measure development/coding/data analysis, running tests, plotting and formatting all graphs and figures

Owen: initial formulation of urbanness, collecting data of original closeness measures, building dual graphs from SNAP graphs, creating the traffic simulator class, creating car's class, all development of traffic simulation and simulation mathematics