KASA New York Traffic Mapping

Abdallah Aboelela, Kei Irizawa, Adam Oppenheimer, Swayam Sinha

June 2019

Description of Datasets

For this project, we used three datasets.

1. 2009 - 2016 TLC New York Taxi and Limousine Commission (TLC) Trip Data
2. Post-November 2014 Vision Zero (VZ) Speed Limit Data
3. OpenStreetMap Graph Data

New York TLC

This dataset was available in two places. It was publicly available on Google BigQuery, and has also been uploaded as 800 separate parquet[1] files on Academic Torrents.[2]. Though we attempted to use BigQuery to query the full dataset and convert to csvs ourselves, we found that due to the size of the dataset, it would be much better to torrent from Academic torrents (done once), and then convert to csvs and upload to Google Cloud Storage. We have two separate buckets in GCS for the data in its original parquet form, and for data in its converted form.

The 800 files are numbered from 0 to 799 and ordered by the datetime of the first trip in each file. Each file contains approximately 1.7 million taxi trips. Each trip is recorded in one entry, and contains data such as pickup and drop off latitudes, longitudes, and datetimes. It also included distance covered, payment types, tip amounts, fare, and a unique id associated with the trip, though we did not use this data. Each parquet file is about 40 MB, and each csv we converted is about 300 MB. In total, the dataset contains about 1.7 M trips $\times$ 800 files = 14.4 Billions trips, which is 300 MB $\times$ 800 files = 240 GB worth of data in csv form.

Speed Limits

Our speed limit data are gathered from the New York Vision Zero (VZ) Speed Limits dataset. This dataset includes speed limits for most streets in New York City. This dataset was downloaded May 8th, 2019. Streets without an explicit speed limit are inferred to be 25 miles per hour. This reflects the change in speed limits for unsigned streets from 30 to 25 miles per hour in November 2014. The dataset was downloaded as a geojson file with a size of 37.2 mb.

Graph

To get graph data for New York City, we first downloaded a polygon file to represent the outline of New York City. The polygon was formatted as a geojson file with a size of 3 mb. We then used OpenStreetMap with the polygon of New York City to retrieve a NetworkX Graph representation of New York City.

General Strategy and Major Milestones

1. Torrent all files from Academic Torrent and upload to GCS (manual)

[1] Parquet is a column based file storage system, which can be read by pandas
[2] https://academictorrents.com/details/4f465810b86c6b793d1c7556fe3936441081992e

2. Convert all files to csv by downloading, reading and rewriting to csv and reuploading to GCS (using gsutil and pandas). This was run on 4 separate Compute Engine VM instances.
3. Change graph edges to use times instead of distances for length
4. Download OSMNX source code to make Dijkstra's Algorithm compatible with MRJob (see Challenges)
5. Generate reference csv containing start and end datetimes for each csv file (used gsutil and pandas, assumed data chronological). The reference contains data for the 800 torrented and converted csvs. This was run on a Compute Engine VM instance.
6. Choose which dates to run in MRJob (We decided on holidays which occur on the same day each year, as well as a generic week) and download the relevant files for each year using the reference file from the previous step as a guideline
7. Create traffic maps based on output from previous step

Big Data Approaches

MRJob and Google Dataproc

In order to overcome the challenges of working with big-data, we chose to run MRJob with its Dataproc option which creates a cluster of instances that run separate mappers and combiners. Our preference was to use 4 instances.

MRJob and OSMNX Algorithm

For each taxi trip, we generated the shortest route taken using Dijkstra's algorithm. We represented the routes in terms of a network of nodes, where each node represents the intersection of streets. Therefore, two nodes can represent a street. Instead of representing length of street in terms of distance, we represented it in terms of time taken to drive on the street given speed limits. So, for each trip, we collected pairs of nodes and computed the total delay (actual time spent on trip/perfect trip time from OSMNX) for each node pair and time of day, and took an average of the delay for each unique node pair, time of day, year combination. In our output csvs, each node pair appears up to 4 times per year, once for each time of day.

Runtime

It took approximately 25 hours to run the entire program for all of our chosen dates through Google Dataproc on a cluster of 4 instances. If we run on one single machine we extrapolate to take about 100 hours.

Challenges Faced

There were three major challenges that we faced.

Challenge 1: Djikstra's Time vs. Distance

The goal of the project was to construct a map that describes streets with high congestion. However, our taxi data set only provides us with the pickup and drop-off locations, giving us no information on the path it took to get there. Therefore, we make an assumption that the taxi takes the shortest path possible from pickup to drop-off. We apply Dijkstra's algorithm to find this shortest path, with the length of the path being dependent on time, not distance. We found the speed limit for each street and used it to calculate the time it takes for a car to go through that street assuming it travels at the speed limit.

Challenge 2: Djikstra's Run Time

For each trip, our version of OSMNX's Djikstra's ideal path algorithm took about 2 seconds to run. Given that there are about 2 million taxi trips per day we looked at, this would take an unreasonably long time to run. Instead, we decided to sample 1% of the trips in each day we looked at, and used Google Dataproc to speed up our runtime by running on 4 instances at the same time.

Challenge 3: OSMNX Import

For some reason that we are still unable to figure out, Google's instances refused and continue to refuse to install OSMNX. This was incredibly frustrating because while our code ran fine locally, we were not able to run it in dataproc. Instead, we found downloaded the OSMNX source code from GitHub and copied the relevant functions into util.py and used them as such.

Small Challenge 1: Time

We used time to drive down streets to represent length of streets for Dijkstra's Algorithm. However, this is based on speed limits, which change over time. The speed limits we use represent the post-November 2014 reduction in unsigned speed limits from 30 to 25 miles per hour. However, our data spans a period that crosses this change. Unfortunately, because we have no way to know which streets changed from a speed limit of 30 miles per hour to 25 miles per hour, and which were 25 miles per hour before the change, we chose to keep speed limits constant for all dates. This causes some anomalies in our results, including many cases where the actual time for the ride is less than the lower threshold given by Dijkstra's Algorithm. This is likely caused by cars following a speed limit higher than our data indicates (although the possibility that this could be caused by speeding cannot be totally disregarded).

Questions and Discussion

We chose to look at Christmas Eve, Christmas Day, New Year's Eve, New Year's Day, the 4th of July, and April 1st through 7th of each year the data is available.

We highlight maps from 2009 in our appendix and results discussion because it is a year before the rise of ride share. We believe that this will give us a better understanding of the New York's traffic flow as the populations of those who primarily use ride share apps are likely systematically different from those who use taxis and thus may have taken different trips.

Weekdays vs. Weekends

We looked at a generic week (April 1st through April 7th) of each year. Our thought process was that choosing a full week would give us data on every weekday regardless of the year we look at. As such, we look at the mornings and evenings of the first Wednesday, Friday, and Sunday of April 2009.

Midtown busier on weekday mornings

Comparing the maps on the left hand side of pg. 6, we see that Midtown New York is generally busier on Monday and Friday than it is on Sunday, which is to be expected given people are making their way to work on those days, whereas Sunday traffic is more likely to be a later brunch or church crowd.

Sunday dinners

On the right hand side of the same page, the maps all look about the same with the exception of a couple of east-west streets in midtown and downtown on Sunday with heavier than average traffic. These could be due to people taking advantage of the day off and having late lunches or early dinners and then calling taxis home.

Holidays

We created maps for each of New Year's Eve, New Year's Day, Christmas Eve, Christmas Day, and compared them with our generic April week.

Christmas delays

On page 9, we can see that Christmas and Christmas Eve traffic is significantly more delayed than on any of the other Holidays we looked at. We believe this could be due to weather issues or excitement over taking the day off. This could also be a period of heavy visitation from international tourists.

FDR Four Freedoms Park

On the mornings (6am to 12pm) of the different celebrations, we see a distinctly dark red line on the Queensboro bridge headed to FDR Four Freedoms Park. We believe this could be due to events occurring at the park.

Times of Day

For each unique year-day pair, we created 5 maps. One each 6 hour time of day (Morning, Afternoon, Evening, Night), and a composite of the previous four. To see how traffic delays change at different times of the day.

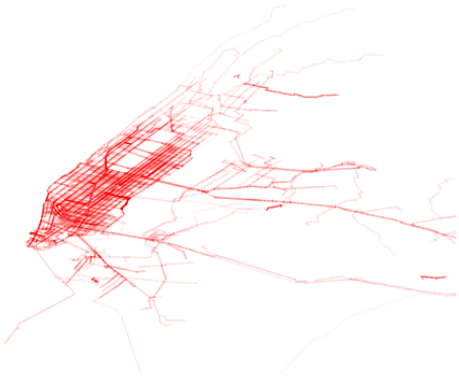
Nighttime construction, Afternoon Rush

Looking at the maps on page 7, in the first few hours of this Wednesday, it appears that there were minor delays spread out across all of New York. We believe this could be due to road work or construction where crews take advantage of the reduced usage to complete work quickly. The afternoon hours of the same day has more concentrated delays in downtown and on the upper west side, likely as people have lunch around noon (the beginning of the afternoon period as defined by us) and head home between 5pm and 6pm (the end of the afternoon period as defined by us).

Results[3]

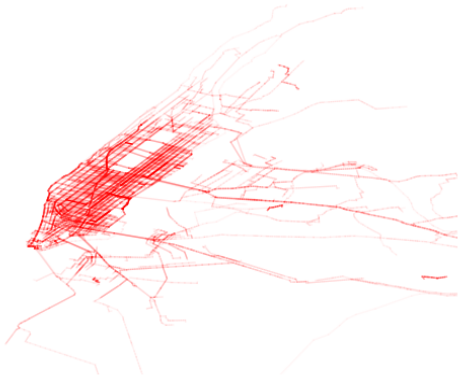
General 2009

april_1st_2009 24 hours



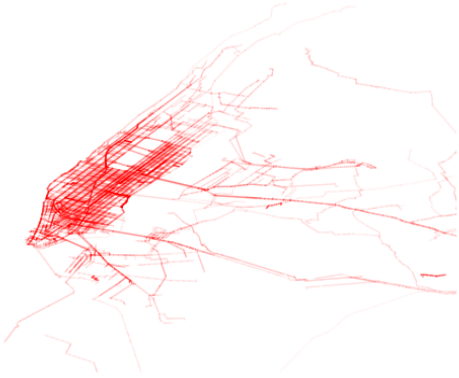
Wednesday

april_2nd_2009 24 hours



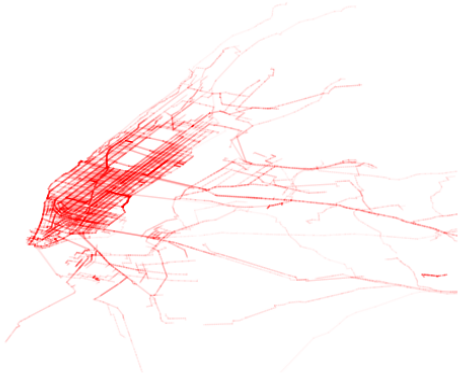
Thursday

april_3rd_2009 24 hours



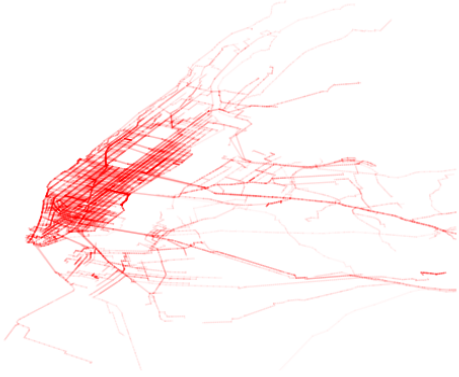
Friday

april_4th_2009 24 hours



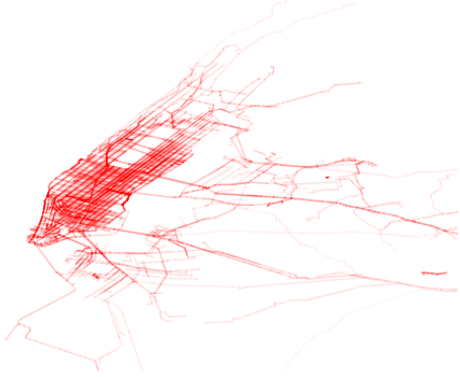
Saturday

april_5th_2009 24 hours



Sunday

april_6th_2009 24 hours

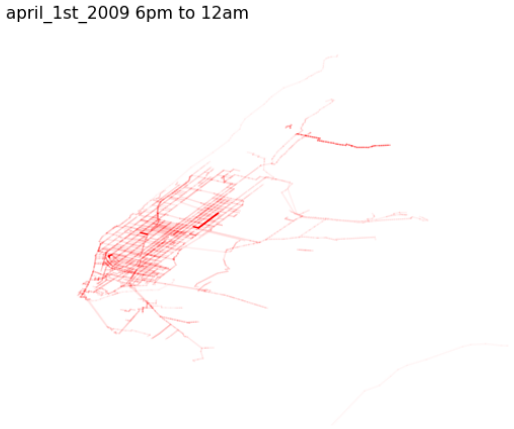


Monday

[3]All maps not included in this report are on GitHub. There are a total of 480 maps. 5 maps per chosen day per year * 8 years * 12 chosen days

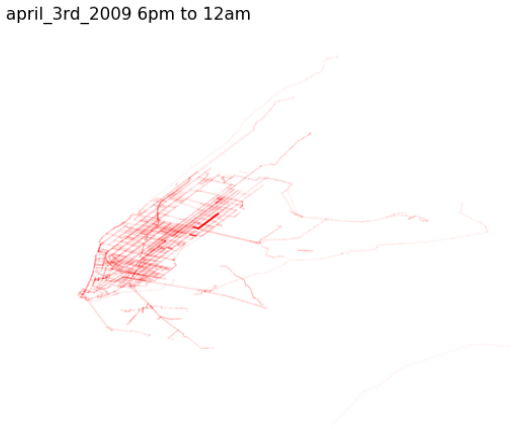
Wednesday, Friday, Sunday Mornings and Evenings

Wednesday Morning



Wednesday Evening

Friday Morning



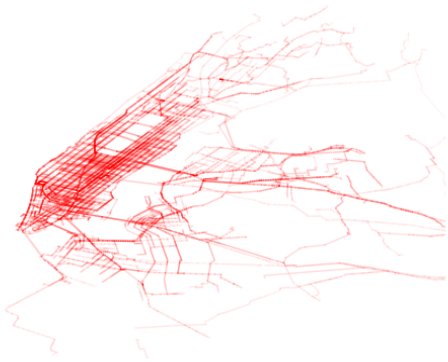
Friday Evening

Sunday Morning

Sunday Evening

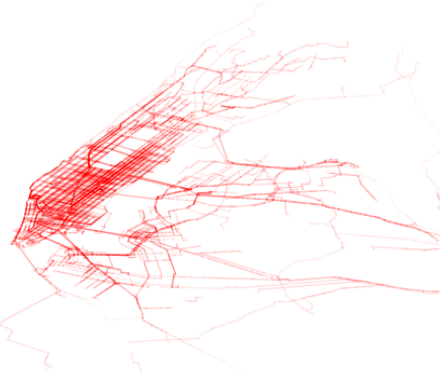
General 2014

april_1st_2014 24 hours



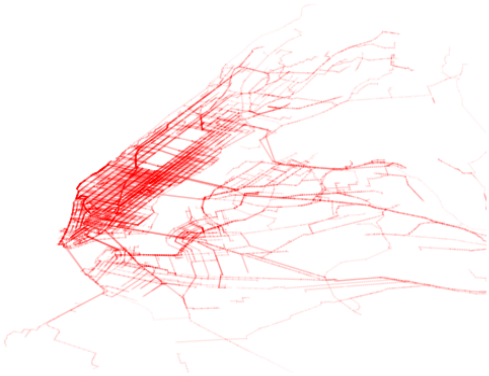
Tuesday

april_2nd_2014 24 hours



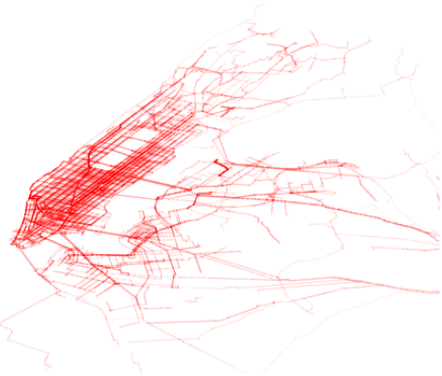
Wednesday

april_3rd_2014 24 hours



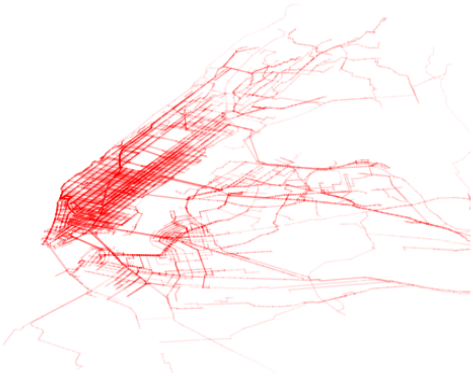
Thursday

april_4th_2014 24 hours



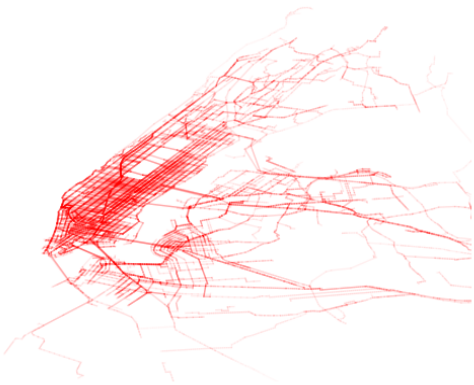
Friday

april_5th_2014 24 hours



Saturday

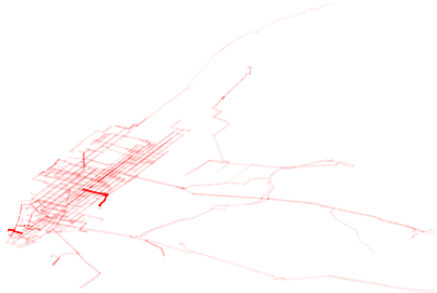
april_6th_2014 24 hours



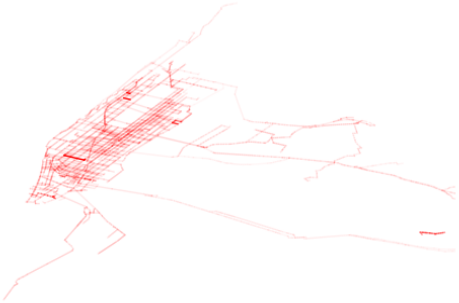
Sunday

General different time (2009)

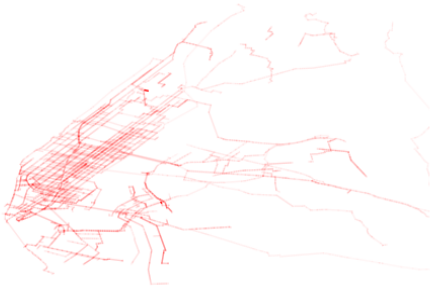
april_1st_2009 6am to 12pm



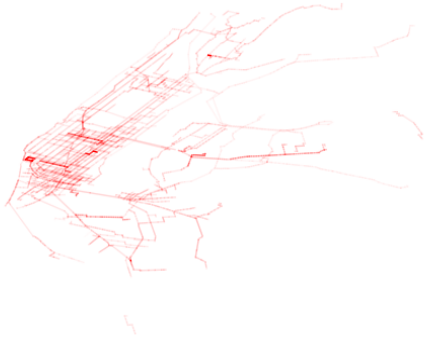
april_1st_2009 12pm to 6pm



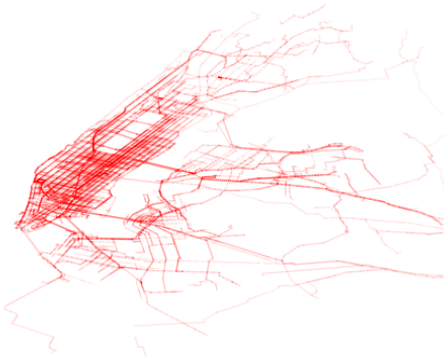
april_1st_2014 6pm to 12am



april_1st_2014 12am to 6am

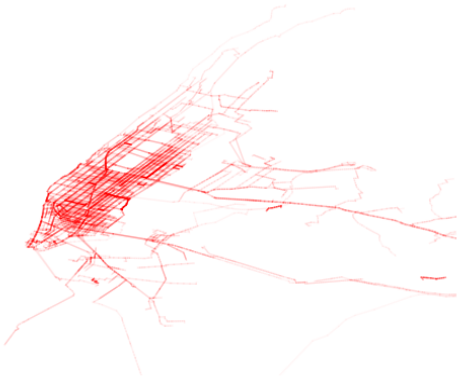


april_1st_2014 24 hours

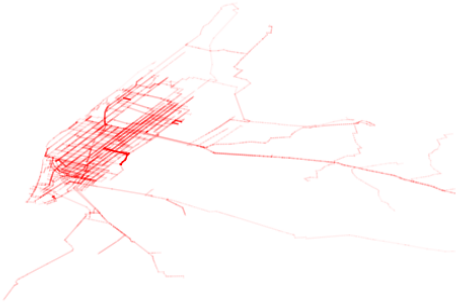


Different Celebration (2009)

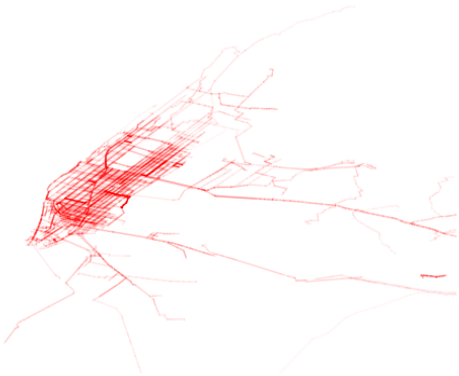
4th_of_july_2009 24 hours



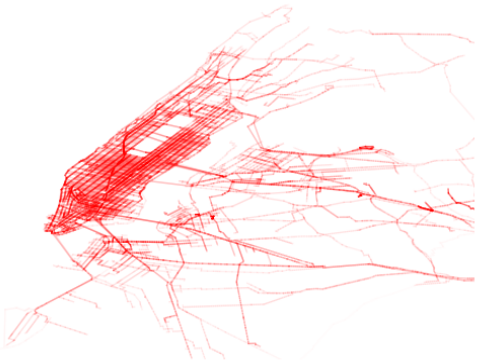
new_years_eve_2009 24 hours



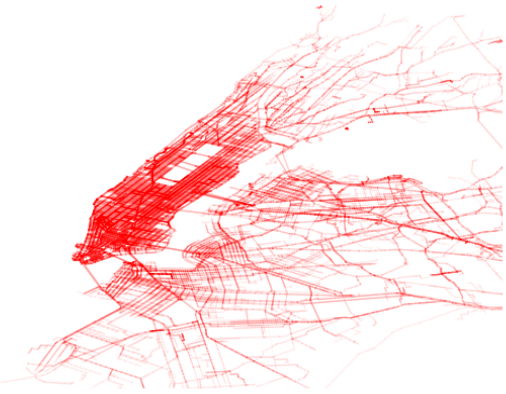
new_years_day_2009 24 hours



christ_eve_2009 24 hours

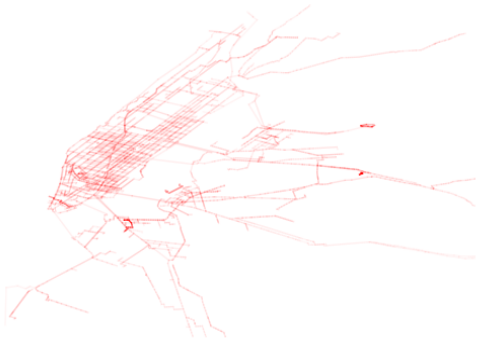


christmas_2009 24 hours

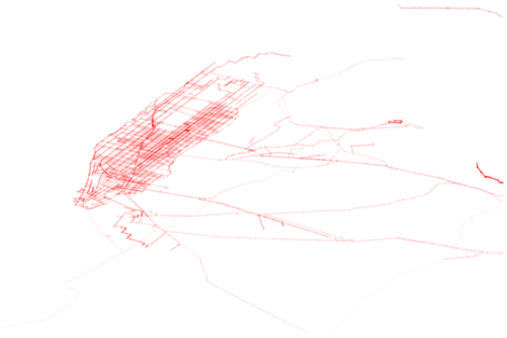


Christmas Eve (2009)

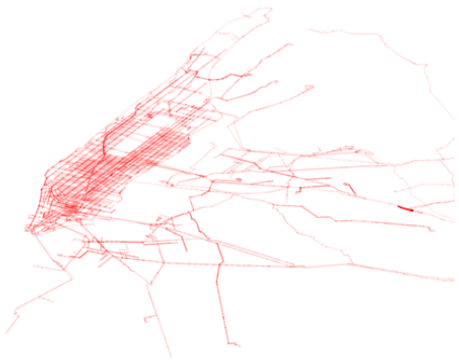
christ_eve_2009 12am to 6am



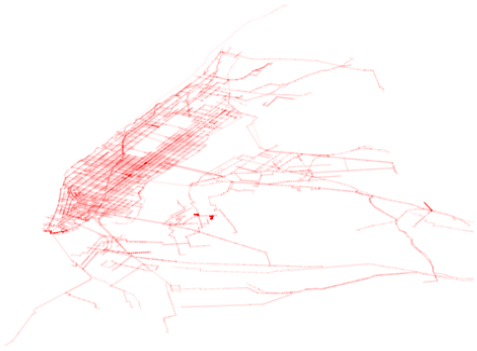
christ_eve_2009 6am to 12pm



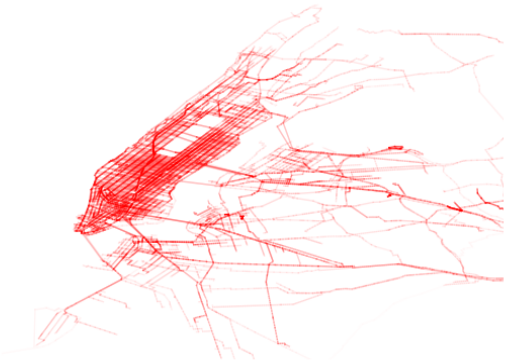
christ_eve_2009 12pm to 6pm



christ_eve_2009 6pm to 12am

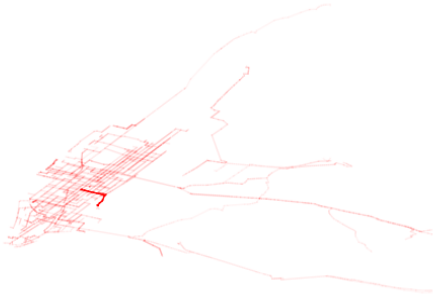


christ_eve_2009 24 hours

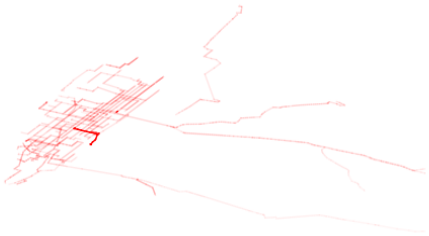


Different Celebration (Morning 2009)

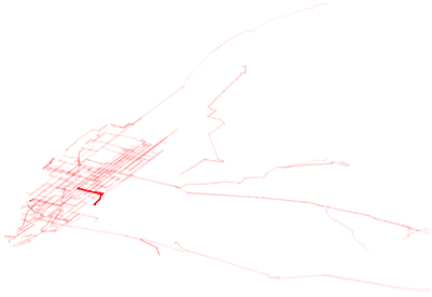
4th_of_july_2009 6am to 12pm



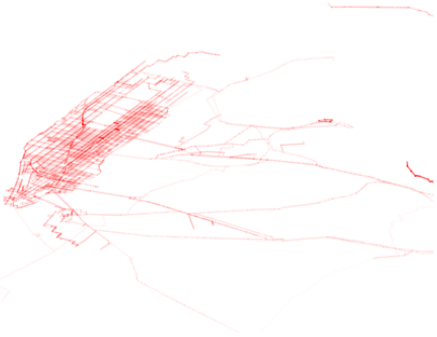
new_years_eve_2009 6am to 12pm



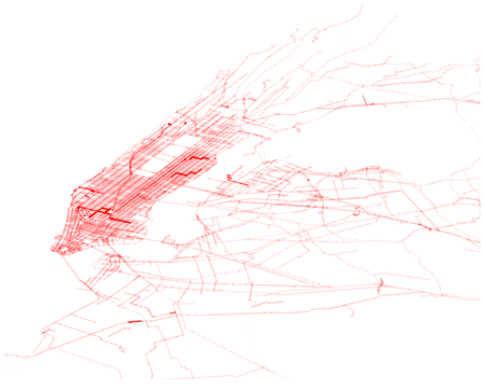
new_years_day_2009 6am to 12pm



christ_eve_2009 6am to 12pm

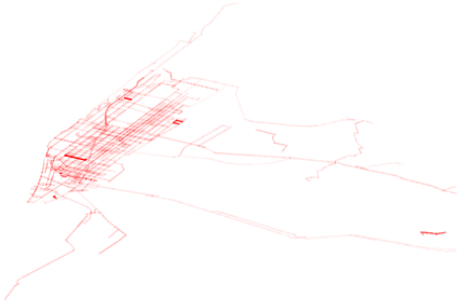


christmas_2009 6am to 12pm

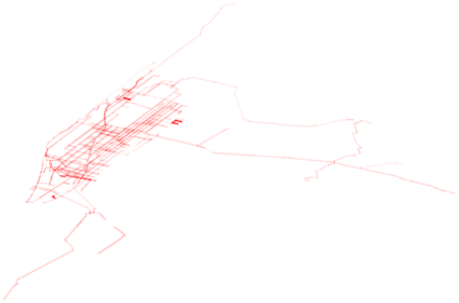


Different Celebration (Afternoon 2009)

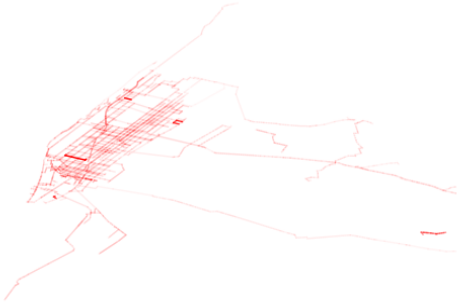
4th_of_july_2009 12pm to 6pm



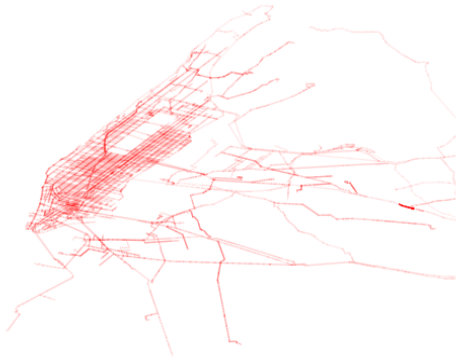
new_years_eve_2009 12pm to 6pm



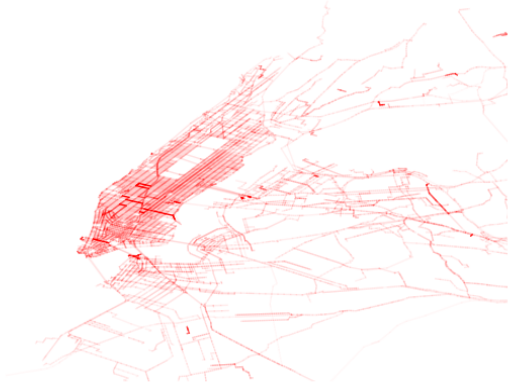
new_years_day_2009 12pm to 6pm



christ_eve_2009 12pm to 6pm

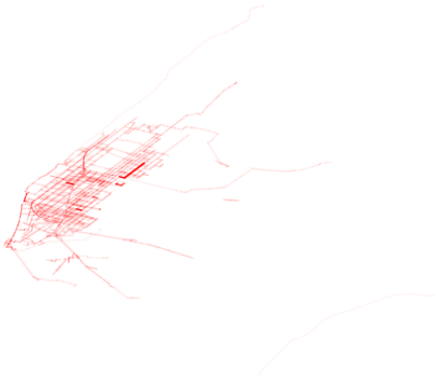


christmas_2009 12pm to 6pm

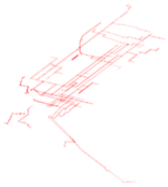


Different Celebration (Evening 2009)

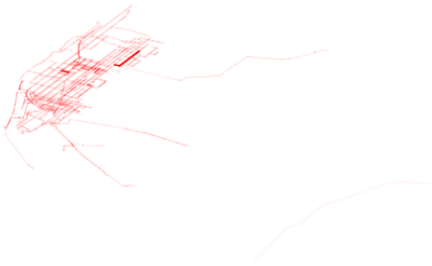
4th_of_july_2009 6pm to 12am



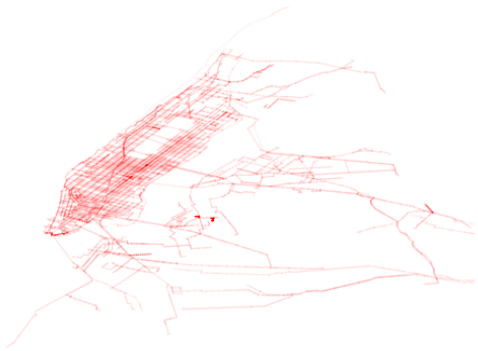
new_years_eve_2009 6pm to 12am



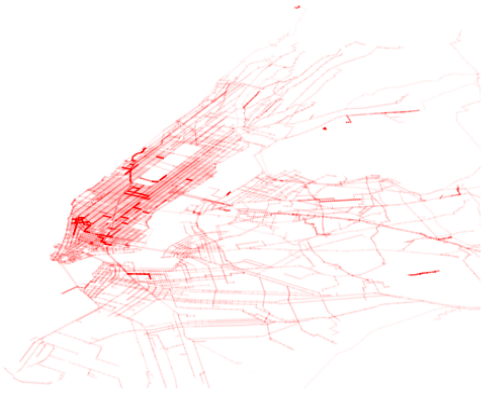
new_years_day_2009 6pm to 12am



christ_eve_2009 6pm to 12am

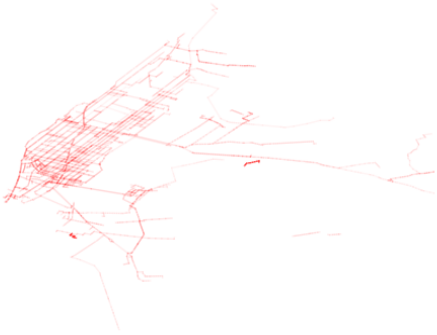


christmas_2009 6pm to 12am

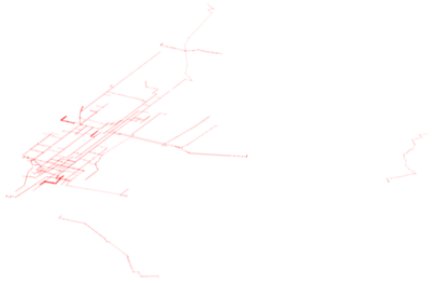


Different Celebration (Night 2009)

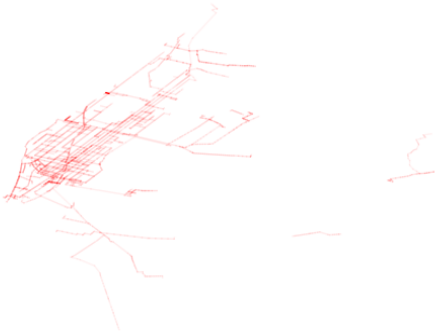
4th_of_july_2009 12am to 6am



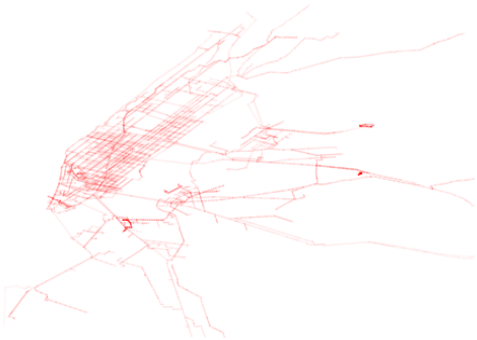
new_years_eve_2009 12am to 6am



new_years_day_2009 12am to 6am



christ_eve_2009 12am to 6am



christmas_2009 12am to 6am