Project KASA: Mapping New York's Traffic

Abdallah Aboelela, Kei Irizawa
Adam Oppenheimer, Swayam Sinha

Taxi & Limousine Commission

Timeline: 2009 - 2016

Available From:

- Google BigQuery (SQL Queries)
- Academic Torrents (Parquet files)

Description:

- Trip level data (pickup and dropoff latitude, longitude, and datetime object)
- Other info (fare amount, method of payment, tip amount)

Size:

- Contains about 1.7 M trips x 800 files = 14.4 Billions trips
- File Size: 300 MB x 800 files = 240 GB

Our purposes:

- Torrented 800 parquet files, converted to csv, and uploaded to GCS

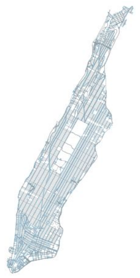
VISION ZERO
nyc.gov/visionzero

Timeline: Post- November 2014

Description:

- Speed limit data for New York Streets

OSMNX



Description:

- Used a polygon outline of New York City used in coordination with OSMNX to create a geojson file (converted to geopandas df object)
- Edited distances in geopandas df to reflect lengths between nodes in terms of time instead of in terms of distance

Dijkstra's Algorithm

Purpose: to generate the path taken by the taxi, operating under the assumption the shortest path was taken.

Implementation:

1. Change length of graph edges to be based on time, rather than distance, using the speed limit data.
2. Use networkx’s ‘shortest_path’ function to generate the route using Dijkstra's algorithm.

Strategy

1. Torrent NYC TLC trip level data from Academic Torrents and upload to GCS
2. Download, read parquet file and rewrite to csv for each of 800 torrented files. Reupload to GCS
3. Change graph edges to use times in stead of distances for length (times calculated by using speed limit data from vision zero and existing distance data from OSMNX graph)
4. Generate reference csv containing first and last pickup datetime object for each of the 800 TLC trip files
5. Choose dates to run in MRJob (Christmas and Eve, New Year's and Eve, 4th of July, Generic week of April) to output unique node pair, year, time of day keys with average delay values
6. Use OSMNX graph and average delays to create maps using Matplotlib

Single Step MRJob

1. Mapper init
    1.1. Open and store G file (time intensive and only needs to be done once per mapper)
2. Mapper
    2.1. Use OSMNX to get ideal path and path time
    2.2. Multiple yields for each node pair in the path
        2.2.1. Key: (year, node1, node2, time of day)
        2.2.2. Value: Average delay of actual trip vs. time of ideal trip
3. Combiner and Reducer
    3.1. Take average of the above

Hypotheses - Generic Week

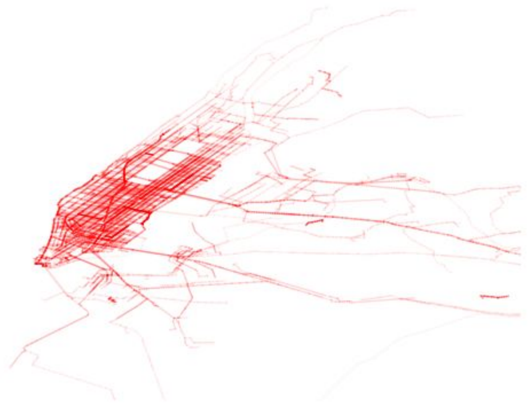
- Consistent Traffic

Generic Week

april_1st_2009 24 hours

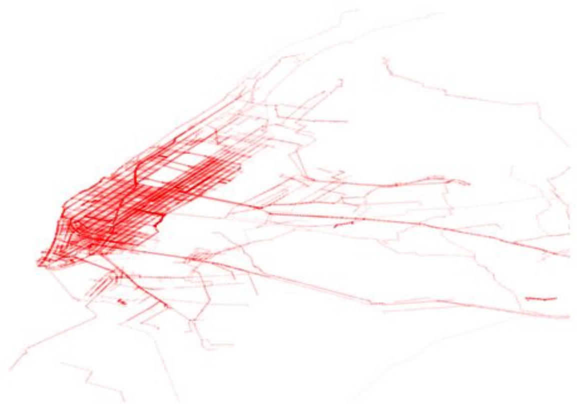
Wednesday

april_2nd_2009 24 hours

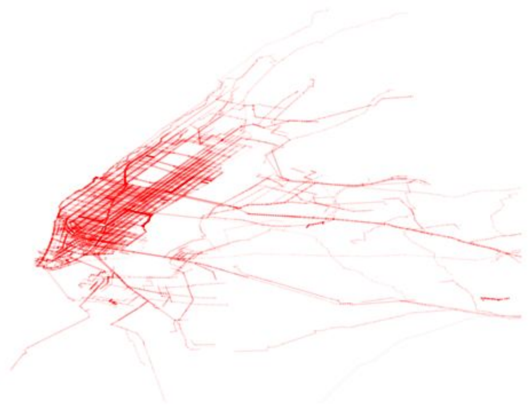


Thursday

april_3rd_2009 24 hours



april_4th_2009 24 hours

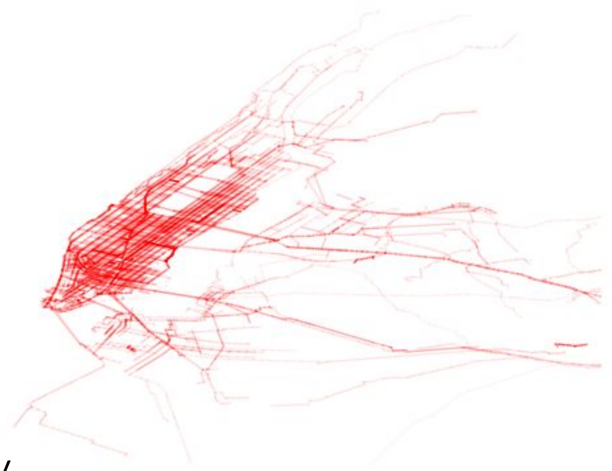


Friday

Saturday

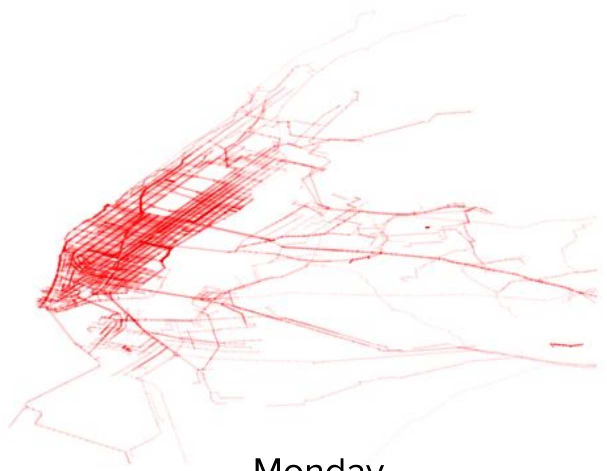
Generic Week

april_5th_2009 24 hours



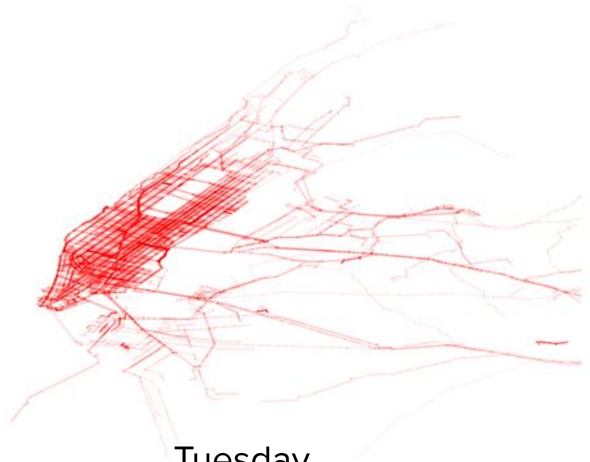
Sunday

april_6th_2009 24 hours



Monday

april_7th_2009 24 hours

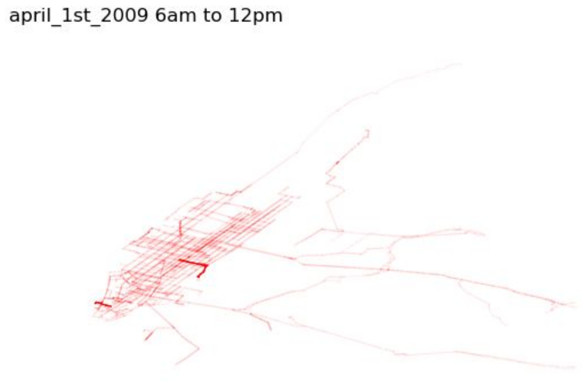


Tuesday

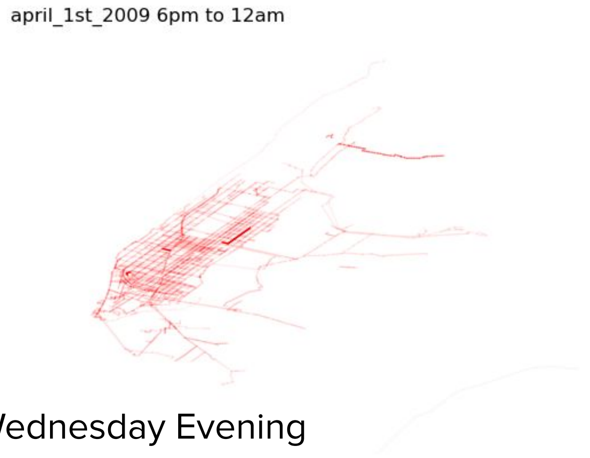
Hypotheses - Weekdays vs. Weekends

- Midtown busier on weekday mornings
- Sunday dinners

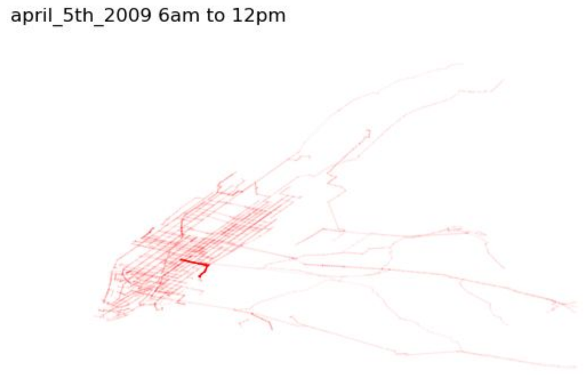
Hypotheses - Weekdays vs Weekends



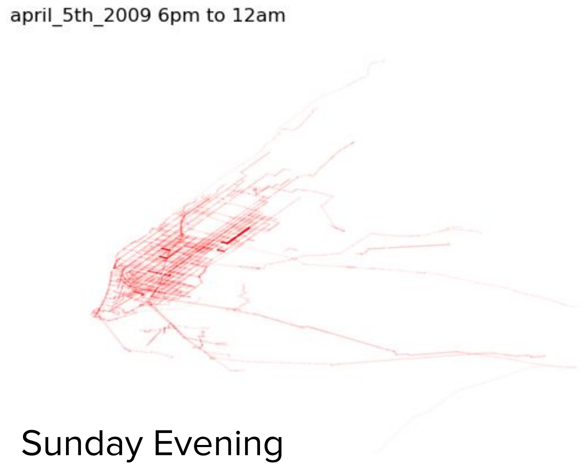
Wednesday Morning



Wednesday Evening



Sunday Morning



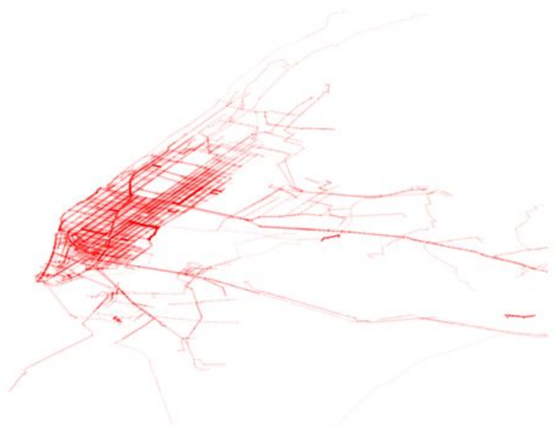
Sunday Evening

Hypotheses - Holidays

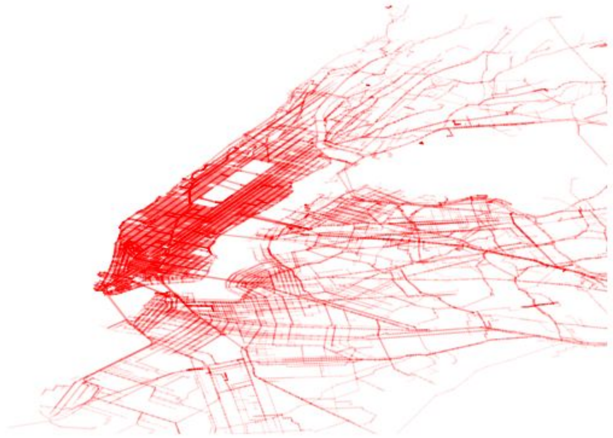
- Christmas delays
- FDR Four Freedoms Park

Holidays

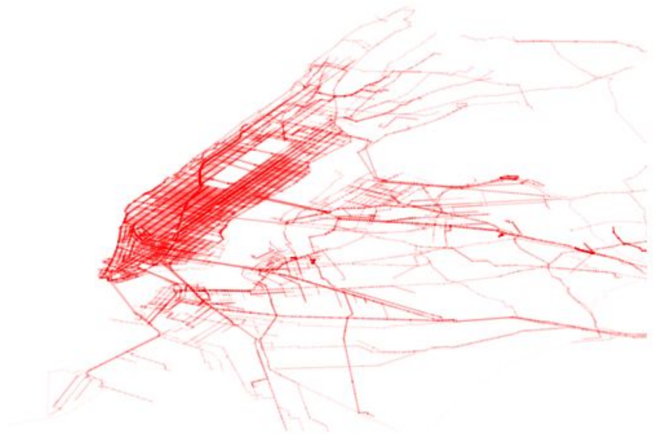
4th_of_july_2009 24 hours



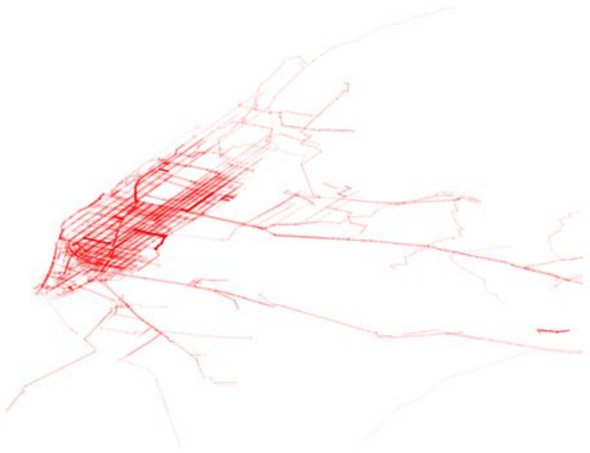
christmas_2009 24 hours



christ_eve_2009 24 hours

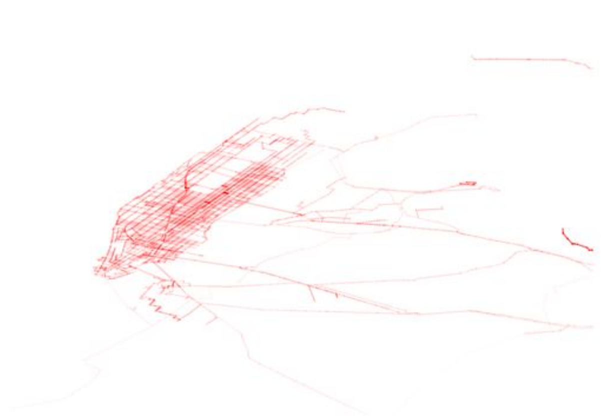


new_years_day_2009 24 hours

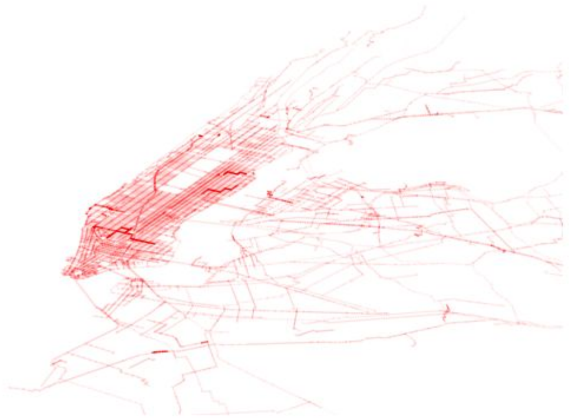


Holidays - Morning

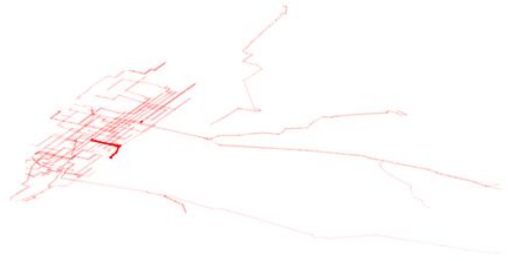
christ_eve_2009 6am to 12pm



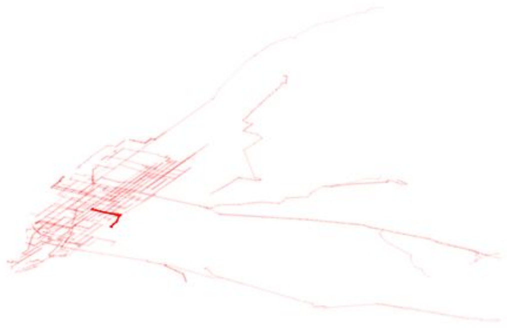
christmas_2009 6am to 12pm



new_years_eve_2009 6am to 12pm



new_years_day_2009 6am to 12pm

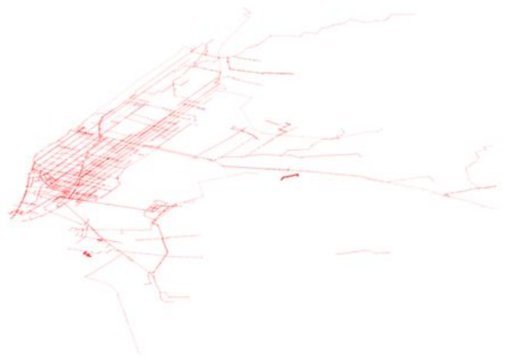


Hypotheses - Times of Day

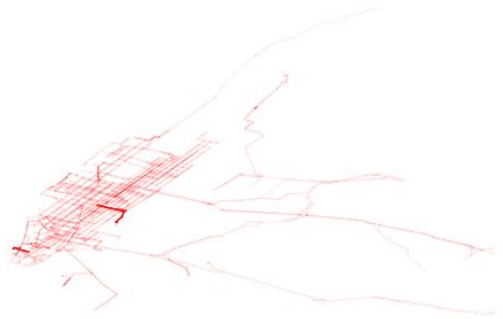
- Night-time construction
- Afternoon rush

Hypotheses - Times of Day

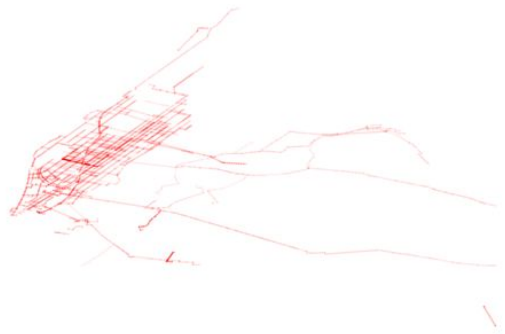
april_1st_2009 12am to 6am



april_1st_2009 6am to 12pm



april_1st_2010 12pm to 6pm



april_1st_2009 12am to 6am