CSP 554 Final Project

Big Data Processing pipelines using Real time Traffic data,Kafka, Spark streaming and Cassandra for Traffic Data Monitoring

Submitted by:

Geethanjali Vivekanandan (A20405203)

Tina Lekshmi Kanth(A20411284)

INDEX

Introduction:

Traffic congestion poses a challenge whenever the demand for transportation system is outspaced by its capacity.It contributes to typical day-to-day problems like late appointments, wear-and-tear of vehicles and emotional stress. Studies also suggest that traffic congestion impacts on the economy (in the form productivity loss), the road surface integrity and the environment.

Project Objective:

Build big data processing pipeline to visualize real time Chicago traffic data. Data is retrieved from City of Chicago data portal via SODA API.The data is sent to Spark streaming using Kafka which is a high throughput messaging system and create dashboard in Power BI to visualize ,Time of the day and levels of congestion based on origins and destination.

Technology:

To begin with we need to analyse the streaming data effectively to get real-time insights, so that we can monitor the changes to find pattern and explore further options.

Fast, flexible, scalable, and resilient data workflow can be made using frameworks like Apache Kafka and Spark Structured Streaming. These systems can be used to process data from multiple real-time sources, process machine learning tasks and experiment effectively with the real-time streams.

Kafka:

Apache Kafka is high-throughput distributed messaging system in which multiple producers send data to Kafka cluster and which in turn serves them to consumers. It is a distributed, partitioned, replicated commit log service.

Spark:

Spark is an open source, fast and general purpose big data processing system for cluster computing. Spark can run in standalone mode, Hadoop YARN or Apache Mesos. To develop applications, Spark provides rich API in Scala, Java and Python. Apart from Spark core engine, Spark comes with several libraries which provides API for parallel computing. The machine in which spark application (Spark Context) runs is called Driver node. Driver executes various parallel operations on worker nodes or cluster. Spark uses concept of a Resilient Distributed Dataset (RDD), which represents a read-only collection of objects partitioned across a set of machines that can be rebuilt if a partition is lost.

Spark Streaming:

Spark Streaming is an extension of Spark Core which provides capabilities of fault tolerant processing of live stream data. Spark streaming divides the incoming stream into micro batches of specified intervals and returns Dstream. Dstream represents continuous stream of data ingested from sources like Kafka, Flume, Twitter, or HDFS. Dstreams are processed and pushed out to file systems, databases, and live dashboards.

Cassandra:

Apache Cassandra is a highly scalable, high-performance distributed database designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. It is a type of NoSQL database.It is an open source, distributed and decentralized/distributed storage system (database), for managing very large amounts of structured data spread out across the world. It provides highly available service with no single point of failure.It is scalable, fault-tolerant, and consistent.column-oriented database.Cassandra accommodates all possible data formats including: structured, semi-structured, and unstructured. It can dynamically accommodate changes to data structures according to the need.

Power BI:

Power BI is a collection of software services, apps, and connectors that work together to turn the unrelated sources of data into coherent, visually immersive, and interactive insights. Whether the data is a simple Excel spreadsheet, or a collection of cloud-based and on-premises hybrid data warehouses, Power BI lets you easily connect to your data sources, visualize what's important, and share the same. Power Bi dashboard is a single page, the visualizations you see on the dashboard are called tiles and are pinned to the dashboard from reports. Power BI reports can be created using Azure cosmos DB connector. They have the function to refresh the data using the scheduler option, that makes this tool dynamic and popular for dashboard application.

Data Source:

alternative routes, crashes, short length of the segments, etc. speed on individual arterial segments can fluctuate from heavily congested to no congestion and back in a few minutes. The segment speed and traffic region congestion estimates together may give a better understanding of the actual traffic conditions. The data reflects the last 10 minutes of traffic and is updated every 15 minutes.The Chicago Traffic Tracker estimates traffic congestion on Chicago's arterial streets (non-freeway streets) in real-time by continuously monitoring and analyzing GPS traces received from Chicago Transit

Authority (CTA) buses. Congestion estimate by traffic segments gives the observed speed typically for one-half mile of a street in one direction of traffic.There is much volatility in traffic segment speed. However, the congestion estimates for the traffic regions remain consistent for relatively longer period. Most volatility in arterial speed comes from the very nature of the arterials themselves. Due to a myriad of factors, including but not limited to frequent intersections, traffic signals, transit movements, availability of

Workflow:

Data Extraction:

Identify a reliable data source, in this case we use data from city of chicago website, next step includes getting SODA API keys and then using Python library called sodapy to connect to the Streaming API and downloading the data.

Data Streaming and Pre-processing:

The data has a predefined schema and is available in a specified format. It is formatted and structured.

Kafka is the tool most people use to read streaming data like this. It follows a publish-subscribe model where you write messages (publish) and read them (subscribe). Messages are grouped into topics. As messages are consumed, they are removed from Kafka. Data is streamed using Spark streaming.

Since we are streaming data it makes more sense if we use streaming product like apache spark.streaming data has value when it is live, i.e., streaming. So there would not be much reason to store that data permanently to some place like Hadoop.

For purpose of traffic data-monitoring, we will be storing the following information on cassandra database,

- Street Name.
- Start and End of traffic congestion, Length
- Direction of traffic
- Current speed
- Time of day, week and month.

Data Analysis:

This step essentially includes how we visualize the data.For this use case data from cassandra DB is queried to display top ten congested route, day and time of most traffic congestion.This will help us speculate what can be possible cause for these results. The in detail analysis of these results will require more data from different sources to validate our speculations, it is not in the scope of this project.This project emphasizes more on building the big-data pipeline to monitor traffic congestion.

Data Visualization:

Like discussed above we query cassandra to identify and display top ten congested route, day and time of most traffic congestion. To do this we need a dashboard kind of visualization. For this purpose we use Power BI application which will retrieve data from azure cosmos DB and send it to Power BI app. We need to ensure that the data is pushed to the application at fixed intervals in order to refresh it automatically.

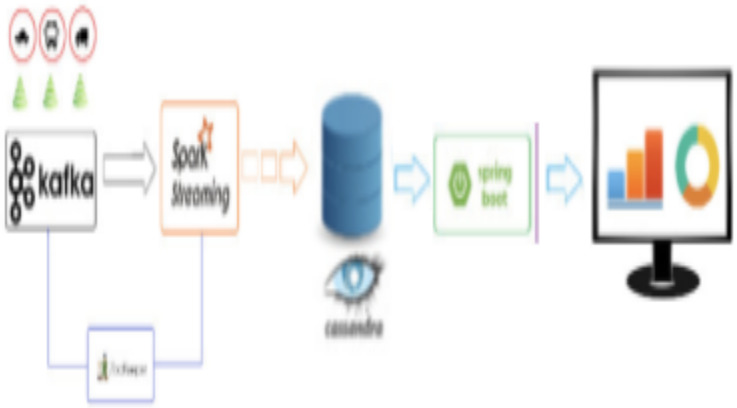


Figure 1. Traffic Data Monitoring Application Architecture Diagram

Process:

For the ease of integration of pipeline Microsoft Azure platform is chosen. Azure provides various tools which can be used to find creative solutions for the given objective. In this case HDInsight is a cost-effective, enterprise-grade service for open source analytics which can be used to run popular open source frameworks like Kafka and Spark.

We use HDInsight tools to create this pipeline along with Azure Cosmos DB.

1. Create the Kafka Topic :

   Identify the Zookeeper host information to use for the kafka cluster. We use azure powershell for this purpose.

2. Retrieve the Traffic data :

   The Traffic data as discussed above, needs to be retrieved from City of Chicago website. Using the Rest API link we retrieve the data onto an array.

3. Set the Kafka broker hosts information:

   Once the Kafka broker information is set for the kafka clusters, we can use this to write data to cluster. Kafka Topic “Trafficdata” is created in this stage.

4. Streaming the data to Kafka:

   Kafka producer is created at this stage and iterate over the data to emit to Kafka.

5. Spark Structured Streaming to retrieve data from Kafka:

   It uses the Azure CosmosDB Spark Connector to write to a Cosmos DB SQL API database.Start the spark session, set the Kafka broker hosts information and the topic.

6. Configure the Cosmos DB connection information:

   Create a database and collection, then retrieve the endpoint, master key, and preferred region information.

7. Cassandra API configuration in Spark2:

   The Spark connector for Cassandra requires that the Cassandra connection details to be initialized as part of the Spark context. Use cassandra connector and datastax Connector to create keyspace and table.

8. Download the data into Cassandra API and write to cosmos DB:

   Define a schema (generally follows the pattern of the json file of data) and add the data to Azure cosmos DB.

9. Connect Power BI and Azure cosmos DB to create the dashboard:
   - Run Power BI Desktop.
   - You can Get Data, see Recent Sources, or Open Other Reports directly from the welcome screen.
   - Select the "X" at the top right corner to close the screen.
   - The Report view of Power BI Desktop is displayed.
   - Select the Home ribbon, then click on Get Data. The Get Data window should appear.
   - Click on Azure, select Azure Cosmos DB (Beta), and then click Connect.
   - On the Preview Connector page, click Continue. The Azure Cosmos DB window appears.
   - Specify the Azure Cosmos DB account endpoint URL to retrieve the data and then click OK.
   - Retrieve the URL from the URI box in the Keys blade of the Azure portal.
   - When the account is successfully connected, the Navigator pane appears. The Navigator shows a list of databases under the account.
   - Populated data from Cosmos Db to create the dashboard in Power BI and is scheduled for refresh.

10. Creating Power BI Dashboard for Visualization:

    - Dashboard is created in Power BI using real time traffic data streamed from Microsoft Azure.
    - Power BI is set up for scheduled refresh to populate the most updated data.
    - A new column, Category is created to be legend in the dashboard for better appeal and ease of understanding. The column is created based on length of traffic into “Easy to go”, “Light” and “Heavy” , using Query editor in Power BI.
    - If the traffic length is less than ¼ th of the average traffic length , the route will indicate “Easy to go” traffic and if the traffic length is above average the route will indicate “Heavy” traffic and anything in between will be indicated as “Light”.
    - Power Query:

```
Category = IF(TrafficData[Length]<(AVERAGE(TrafficData[Length])/4),"Easy to go",IF(TrafficData[Length]>(AVERAGE(TrafficData[Length])/4),"Light",IF(TrafficData[Length]>AVERAGE(TrafficData[Length]),"Heavy")))
```

- We have included From_Street column and Direction as slicers.
- A user can go to dashboard and chose origin from “From_Street” and direction he wishes to commute from “Direction”.
- The dashboard will show the traffic length to various “To_Streets” ,which are destination streets in the chosen direction from origin. Since the column category is legend to dashboard, its color coded and easy to see whether the traffic is “Easy to go “ ,”Light” or “Heavy”.

Challenges:

1. Data is gathered by City of Chicago from CTA buses and hence the traffic data may not reflect the exact scenario in the segment.
2. Few segments have less coverage and hence data is not updated for every 15 minutes in such segments.
3. Azure provides Cassandra API to Cosmos DB with which data can be loaded but for visualization we need to send the stored data to Power BI which allows only SQL API and hence the data had to be loaded again to Cosmos DB using SQL API.

Citations:

1. “Chicago Traffic Tracker - Congestion Estimates by Segments | City of Chicago | Data Portal.” *Chicago*, data.cityofchicago.org/Transportation/Chicago-Traffic-Tracker-Congestion-Estimates-by-Se/n4j6-wkkf.

2. GundConfigure the Cosmos DB connection information:a, Sneha. “Azure Cosmos DB Documentation - Tutorials, API Reference.” Microsoft Docs, docs.microsoft.com/en-us/azure/cosmos-db/.
3. Baghel, Amit. “Traffic Data Monitoring Using IoT, Kafka and Spark Streaming.” InfoQ, InfoQ, 28 Sept. 2016, www.infoq.com/articles/traffic-data-monitoring-iot-kafka-and-spark-streamin

4. Anagha-microsoft. (n.d.). Access Azure Cosmos DB Cassandra API from Azure Databricks. Retrieved from https://docs.microsoft.com/en-us/azure/cosmos-db/cassandra-spark-databricks
5. Azure Samples. (n.d.). Retrieved from https://github.com/Azure-Samples
6. Why Big Data Matters for Traffic Congestion Studies. (2018, November 08). Retrieved from https://www.streetlightdata.com/why-big-data-matters-for-traffic-congestion-studies

Appendix

1. Demo Video: The link contains two videos added as playlist. https://www.youtube.com/watch?v=eyQmzymPLvk&list=PLFB0R1uispVxQ6UGIuJZzMQbfBHJV3xo1
2. Python Notebooks: https://github.com/tinalekshmikanth/Big-Data-Technologies
3. Sample Data in Json:

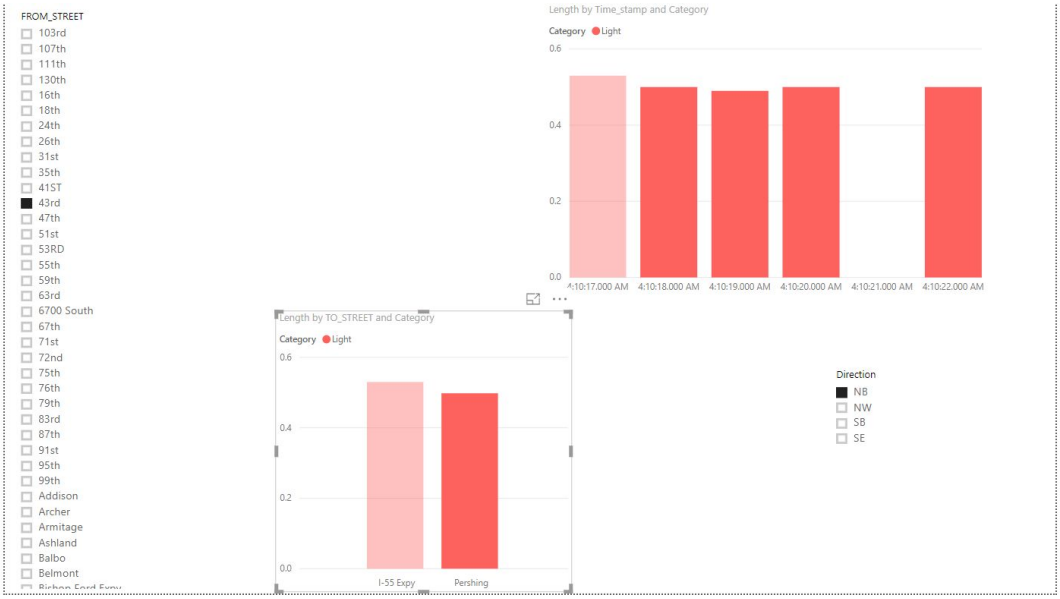
[{"_direction":"EB","_fromst":"Pulaski","_last_updt":"2018-12-01 06:10:22.0","_length":"0.5","_lif_lat":"41.7930671862","_lit_lat":"41.793140551","_lit_lon":"-87.7136071496","_strheading":"W","_tost":"Central Park","_traffic":"23","segmentid":"1","start_lon":"-87.7231602513","street":"55th"}
,{"_direction":"EB","_fromst":"Dan Ryan Expy","_last_updt":"2018-12-01 06:10:22.0","_length":"0.27","_lif_lat":"41.7943974487","_lit_lat":"41.7945059819","_lit_lon":"-87.625702556","_strheading":"W","_tost":"State","_traffic":"31","segmentid":"10","start_lon":"-87.6311283131","street":"Garfield"}
,{"_direction":"NB","_fromst":"87th","_last_updt":"2018-12-01 06:10:23.0","_length":"0.5","_lif_lat":"41.7357887154","_lit_lat":"41.7430756949","_lit_lon":"-87.6630683582","_strheading":"S","_tost":"83rd","_traffic":"23","segmentid":"100","start_lon":"-87.6628819511","street":"Ashland"}
,{"_comments":"outside city limits","_direction":"EB","_fromst":"Crawford","_last_updt":"2018-12-01 06:10:37.0","_length":"1.0","_lif_lat":"42.0115985088","_lit_lat":"42.0117196518","_lit_lon":"-87.7090567368","_strheading":"W","_tost":"Kedzie","_traffic":"-1","segmentid":"1000","start_lon":"-87.7285988897","street":"Touhy"}

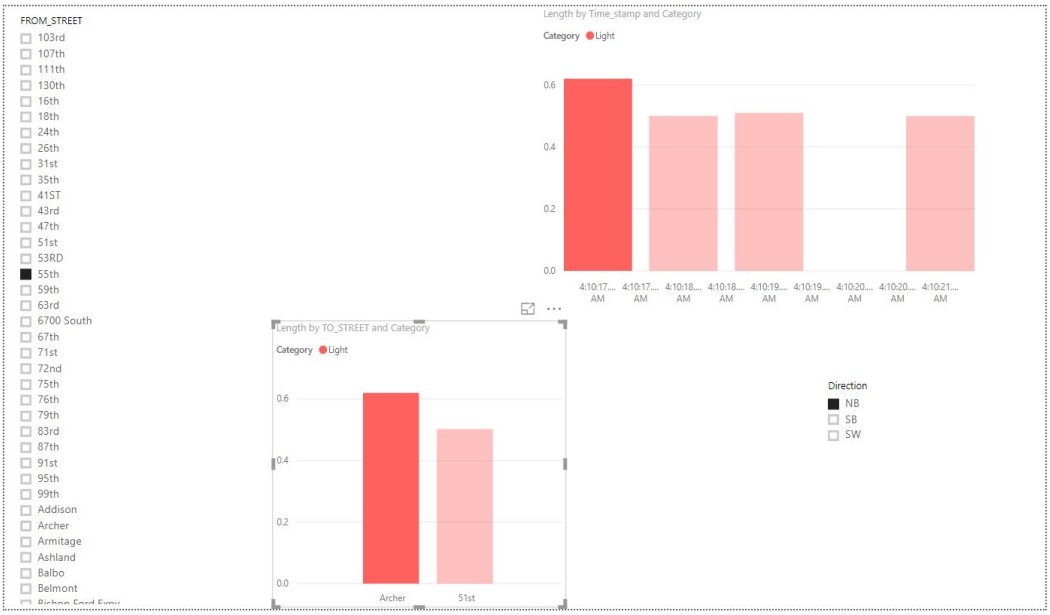
Metadata

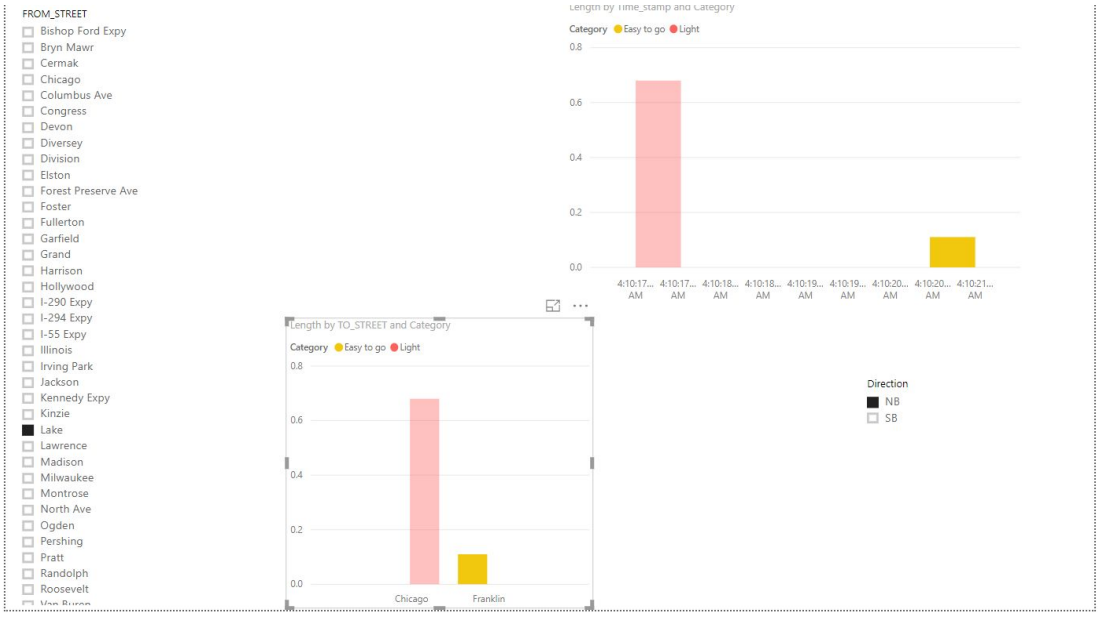
Column Name	Description
SEGMENTID	Unique arbitrary number to represent each segment
STREET	Street name of the traffic segment

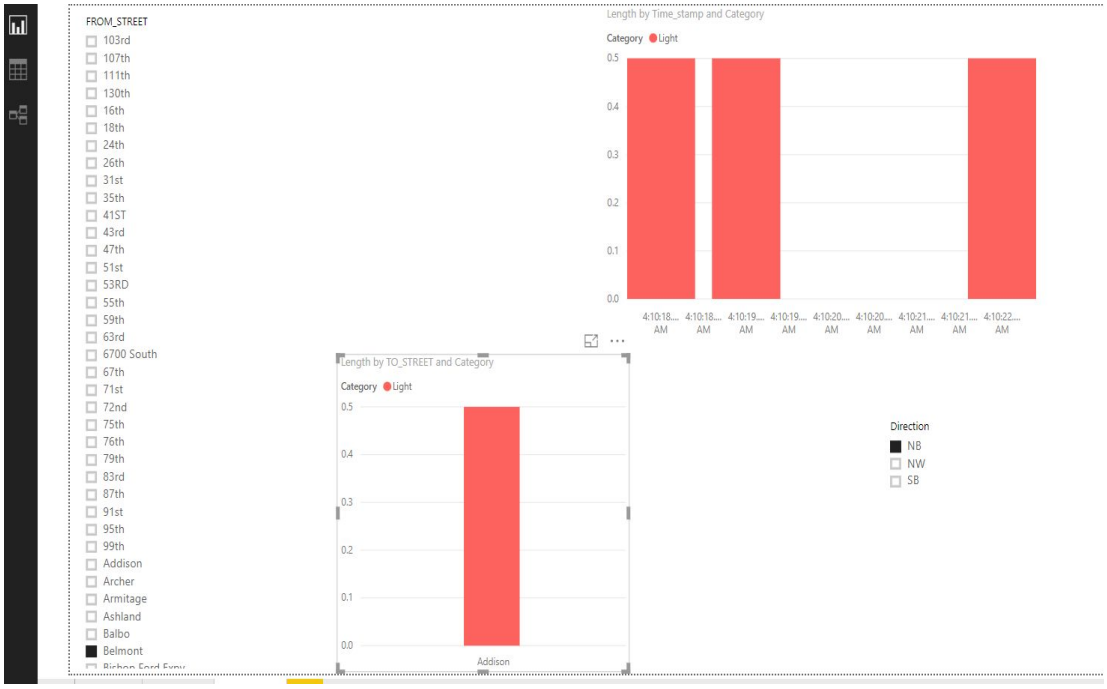
DIRECTION	Traffic flow direction for the segment
FROM_STREET	Start street for the segment in the direction of traffic flow
TO_STREET	End street for the segment in the direction of traffic flow
LENGTH	Length of the segment in miles
STREET_HEADING	Direction of the “STREET” from the origin point
START & END LONGITUDE & LATITUDE:	These four points represent the start and end points of the segment in the direction of traffic flow.
CURRENT_SPEED	Real-time estimated speed in miles per hour
LAST_UPDATED	Timestamp

4. Screenshots from Power BI dashboard:




FROM_STREET
103rd
107th
111th
130th
16th
18th
24th
26th
31st
35th
41ST
43rd
47th
51st
53RD
55th
59th
63rd
6700 South
67th
71st
72nd
75th
76th
79th
83rd
87th
91st
95th
99th
Addison
Archer
Armitage
Ashland
Balbo
Belmont
Length by Time_stamp and Category
Category Light
0.6
0.4
0.2
0.0
4:10:17... AM
4:10:17... AM
4:10:18... AM
4:10:18... AM
4:10:19... AM
4:10:19... AM
4:10:20... AM
4:10:20... AM
4:10:21... AM
Length by TO_STREET and Category
Category Light
0.6
0.4
0.2
0.0
Archer
51st
Direction
NB
SB
SW


FROM_STREET
Bishop Ford Expy
Bryn Mawr
Cermak
Chicago
Columbus Ave
Congress
Devon
Diversey
Division
Elston
Forest Preserve Ave
Foster
Fullerton
Garfield
Grand
Harrison
Hollywood
I-290 Expy
I-294 Expy
I-55 Expy
Illinois
Irving Park
Jackson
Kennedy Expy
Kinzie
Lake
Lawrence
Madison
Milwaukee
Montrose
North Ave
Ogden
Pershing
Pratt
Randolph
Roosevelt
Length by Time_stamp and Category
Category Easy to go Light
0.8
0.6
0.4
0.2
0.0
4:10:17... AM
4:10:17... AM
4:10:18... AM
4:10:18... AM
4:10:19... AM
4:10:19... AM
4:10:20... AM
4:10:20... AM
4:10:21... AM
Length by TO_STREET and Category
Category Easy to go Light
0.8
0.6
0.4
0.2
0.0
Chicago
Franklin
Direction
NB
SB


FROM_STREET
103rd
107th
111th
130th
16th
18th
24th
26th
31st
35th
41ST
43rd
47th
51st
53RD
55th
59th
63rd
6700 South
67th
71st
72nd
75th
76th
79th
83rd
87th
91st
95th
99th
Addison
Archer
Armitage
Ashland
Balbo
Belmont
Length by Time_stamp and Category
Category Light
0.5
0.4
0.3
0.2
0.1
0.0
4:10:18.... AM
4:10:18.... AM
4:10:19.... AM
4:10:19.... AM
4:10:20.... AM
4:10:20.... AM
4:10:21.... AM
4:10:21.... AM
4:10:22.... AM
Length by TO_STREET and Category
Category Light
0.5
0.4
0.3
0.2
0.1
0.0
Addison
Direction
NB
NW
SB