Incident Detection

Bachelor of Technology
Thesis
by

Amogha Hakkare Arunachala
150040102

under the guidance of

Prof. Gopal R Patil

Indian Institute of Technology, Bombay
Mumbai 400 076

Contents

1 Abstract i

2 Acknowledgement ii

3 Introduction iii

4 Study Area and Data Collection v

5 Distance Measurement vii

6 Twitter Module x

7 Webpage for Data Analysis xiv

8 Google Maps Marker xvii

9 Analysis of data obtained xix

10 Conclusion and Scope of Improvement xxiii

Chapter 1

Abstract

The importance of travel time reliability in transportation systems management, control and network design has been a significant focus and has received a lot of attention in the past decade. What is occurring today is a fundamental shift from a past policy focus on average travel time to one that now focuses on variability of travel time. Most travellers are less tolerant of unexpected delays because such delays have larger consequences than drivers face with everyday congestion. Various agencies are already showing interest and figuring out how to implement travel time reliability- related performance measures, analytical processes, and tools into their planning and programming processes. Foreign agencies such as the Federal Highway Administration (FHWA), Minnesota Department of Transportation (Mn/DOT), and the Washington State Department of Transportation (WSDOT) have already started to use travel time reliability as a performance measure to supplement measures of average congestion. Countries like India which is developing fast with rapid urbanisation and industrialisation need to implement reliability as a performance measure to tackle congestion and jams on roads. The aim of this project is to collect travel time data, analyse and measure the reliability index for 4 major highways of Mumbai city. The data was collected with the help of google maps API which was coded in python. Script scheduling was used to collect data on regular intervals thought the day for nearly 1 month. Reliability measures such as planning time index (PTI), Planning Time (PT) and Buffer Time Index (BTI) were studied. Results revealed that reliability measures are more capable of evaluating the performance of urban roadways. It was found from analysis that certain segments of Western Express Highway had very high reliability indices and Eastern Express Freeway was the most reliable among the 4 highways.

Chapter 2

Acknowledgement

I would like to thank my supervisor, **Prof. Gopal R. Patil** for his inspiration, guidance and insightful suggestions without which this project would have not been possible. The interactive sessions with him definitely helped me clearing my doubts regarding the project and have benefit-ted me with immense knowledge. Also, I would like to thank my friends and seniors who provided valuable feedback for this project.

This acknowledgement would be incomplete without the mention of my parents, whose constant encouragement, love and support have helped me in every step of my life.

Amogha HA
150040102

Chapter 3

Introduction

The urban population of India is cities are increasing at a rate of 2.33 percent per annum. The Mumbai citys car population is bursting, as over past five years alone, the number of vehicles has grown from 2 million to 3 million, at a staggering rate of 50 percent. However, the road length in Mumbai, at around 2,000km hasnt changed significantly during this period. With the burgeoning vehicular population and the almost unchanged road length, travel time has become a crucial parameter for evaluating urban corridors and highways in urban areas. Travel time between a pair of origin and destination is the key information to the user for deciding the time of departure, mode of travel and route. Due to increase in vehicular population, traffic problems like congestion, road-accidents etc. are bound to happen which leads to unpredictable traveltime delays and creates overall decrease in transportation system quality.

In system engineering, reliability is defined The probability of a device performing its purpose adequately for the period of time intended under the operating conditions encountered. In road networks, reliability is defined as the network which can guarantee an acceptable level of service on road traffic even if the function of some link is physically damaged or large amount of travel demand is occasionally generated. In technical terms it is the measure of the dispersion (or spread) of the travel time distribution. Even though the concept of travel time reliability is relatively new, many agencies have already started using it, or are finding ways to incorporate into their model. In view of this, travel time reliability analysis has attracted the attention of researchers because of its importance compared with other network reliability measures such as connectivity reliability and capacity reliability. The use of travel time reliability measures is growing in recent years in various

developed countries like USA and Japan. Agencies such as the Washington State Department of Transportation (WSDOT), Minnesota Department of Transportation (Mn/DOT), and the Federal Highway Administration (FHWA) have primarily used travel time reliability as a performance measure to supplement measures of average congestion. While it is considered as an important attribute for planning the urban transportation systems, it also serves as a measure of road service. It can also be used to measure how effective congestion relief services are.

Travel time reliability measures are relatively new, but a few have proven effective. Various reliability indices which rely on travel time distribution are generally used to quantify travel time reliability. The most effective methods of measuring travel time reliability are 90th 7 or 95th percentile travel times, buffer index, and planning time index. Several statistical measures, such as standard deviation and coefficient of variation, have been used to quantify travel time reliability. The study area for examining the fundamental characteristics of travel time reliability includes the four major roadways of Mumbai city of Maharashtra. These roadways include Western Express Highway, Eastern Express Highway, Eastern Express Freeway and JogeshwariVikhroli Link Road (JVLR). Travel time were taken with the help of Google Maps API. The script created, was scheduled to take travel time data at a regular interval of 20 mins. This was done continuously for nearly 1 month to get enough data for performing analysis and to calculate reliability indices. Further, most reliable and most unreliable segments of these roadways were found.

Chapter 4

Study Area and Data Collection

Characteristics of study area

In the present study four major roadways of Mumbai has been selected as the study area. These roadways include Western Express Highway, Eastern Express Highway, Eastern Express Freeway and Jogeshwari-Vikhroli Link Road (JVLR). These roadways are further divided into 3 to 5 segments according to length of the roadway and major intersections lying on it. These roadways serve as lifelines of transportation in Mumbai and a major volume of traffic flows continuously thought the day. Hence studying the reliability of these highways is very important.

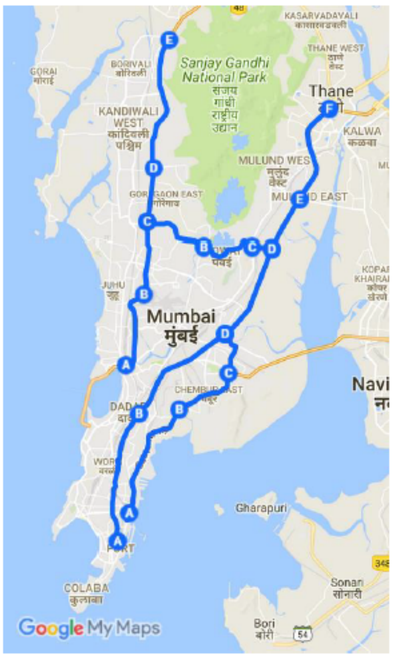


Figure 1 Roadways used in the present study and their segments

NODES	Upstream coordinates	Downstream Coordinates	Remarks
		Eastern Express Freeway	
A	18.9575313, 72.8426089	18.957403, 72.842678	Intersection of EEF with P D' Mello road and SVP Road
B	19.022735, 72.876581	19.022514, 72.876490	Intersection with Sewri-Chembur Road
C	19.045995, 72.909192	19.046171, 72.909342	Shivaji Chowk in Chembur
D	19.070787, 72.907068	19.070877, 72.907216	Intersection with EEH
		JVLR	
A	19.140662, 72.854749	19.140417, 72.854776	Intersection with WEH
B	19.125061, 72.893049	19.124985, 72.893001	Intersection with Saki-Vihar road
C	19.125275, 72.925134	19.125178, 72.925134	Intersection with Lal-Bahadur Sashtri Road
D	19.124104, 72.938513	19.124104, 72.938513	Intersection with EEH
		Western Express Highway	
A	19.050906, 72.840499	19.050701, 72.840564	Southernmost point of WEH; Intersection with SV Road
B	19.095489, 72.852581	19.095472, 72.852675	Node is located near Domestic Airport
C	19.140676, 72.854745	19.140386, 72.854884	Intersection with JVLR
D	19.174343, 72.859353	19.174364, 72.859478	Intersection with Goregaon-Mulund Road
E	19.256670, 72.870892	19.256643, 72.871035	Northernmost point; Intersection with SV Road
		Eastern Express Highway	
A	18.943935, 72.834366	18.943842, 72.834459	Node located near CST
B	19.020245, 72.849537	19.020217, 72.849659	Intersection with Tilak Road
C	19.070785, 72.907089	19.070694, 72.907141	Intersection with EEF
D	19.124099, 72.938574	19.124086, 72.938716	Intersection with JVLR
E	19.154729, 72.956938	19.154699, 72.957086	Intersection with Goregaon-Mulund Road
F	19.211929, 72.977193	19.211842, 72.977245	Northernmost point; Near Thane

Table 1 Major Nodes on the highways being studied

The table above shows the longitudes and latitudes of major nodes on the 4 roadways. Remarks are written which shows the importance of that node on the highway being studied. These are nodes which are taken as a parameter to collect the travel time data. Dividing the roadways into segments is important as it can be used to pinpoint the segments due to which overall reliability of highway changes.

Chapter 5

Distance Measurement

The first and the foremost thing that we want to get is the distance between any two locations. The most accurate method to get the location of a place is to use its latitude and longitude.

Latitude is a geographic coordinate that specifies the northsouth position of a point on the Earth's surface. Latitude is an angle (defined below) which ranges from 0 at the Equator to 90 (North or South) at the poles. **Longitude** is a geographic coordinate that specifies the east-west position of a point on the Earth's surface. It is an angular measurement, usually expressed in degrees and denoted by the Greek letter lambda. **Meridians** (lines running from the North Pole to the South Pole) connect points with the same longitude.

Using **Google Maps API** to find out the exact latitude and longitude of a place is one of the ways we can go about it. After the success of reverse-engineered mashups such as chicagocrime.org and housingmaps.com, Google launched the Google Maps API in June 2005 to allow developers to integrate Google Maps into their websites. It was a free service that didn't require an API key until June 2018 (changes went into effect on July 16), when it was announced that an API key linked to a Google Cloud account with billing enabled would be required to access the API. The API currently does not contain ads, but Google states in their terms of use that they reserve the right to display ads in the future.

The following is the code snippet that we will be implementing. Once the code runs, it'll print out the exact physical address along with the latitude and longitude of the desired place.

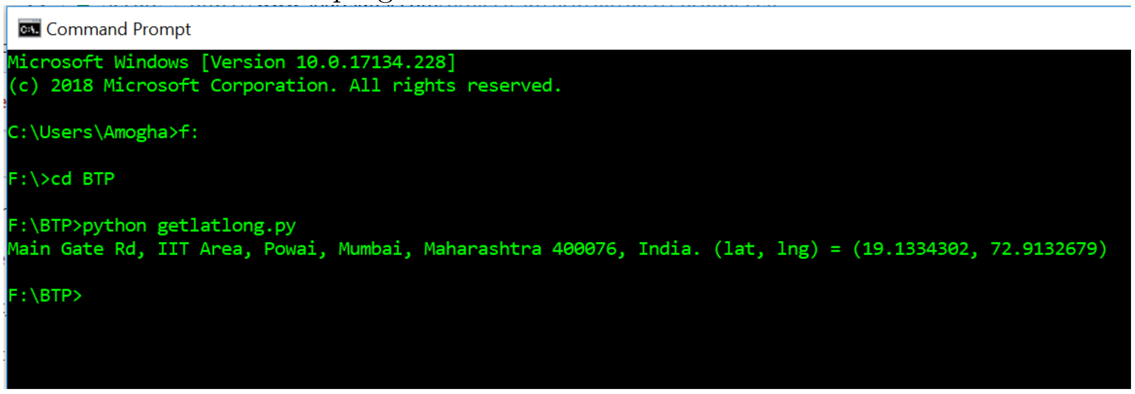
```
import requests

    API_URL = 'http://maps.googleapis.com/maps/api/geocode/json'
    params = {
            'address': 'Powai, IIT Bombay',
            'sensor': 'false',
            'region': 'india'
            }

    req = requests.get(API_URL, params=params)
    res = req.json() # Do the request and get the response data
    result = res['results'][0] # Use the first result
    geodata = dict()
    geodata['lat'] = result['geometry']['location']['lat']
    geodata['lng'] = result['geometry']['location']['lng']
    geodata['address'] = result['formatted_address']

    print('{address}. (lat, lng) = ({lat}, {lng})'.format(**geodata))
```

The result of the above program would be as follows:



```
Command Prompt
Microsoft Windows [Version 10.0.17134.228]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Amogha>f:

F:\>cd BTP

F:\BTP>python getlatlong.py
Main Gate Rd, IIT Area, Powai, Mumbai, Maharashtra 400076, India. (lat, lng) = (19.1334302, 72.9132679)

F:\BTP>
```

The next thing that we have to do is use this latitude and longitude value to find out distance between the two desired locations. We also need to calculate the driving distance between these two locations using the same program. Google Maps API already has a provision in it for the same. The complete project is written in Python 3 (Latest Version). The following is the code that is to be implemented.

```
import googlemaps
from datetime import datetime

gmaps = googlemaps.Client(key='Our Key')
now = datetime.now()
directions_result = gmaps.directions("18.997739, 72.841280",
                                     "18.880253, 72.945137",
                                     mode="driving",
                                     avoid="ferries",
                                     departure_time=now
                                    )
print(directions_result[0]['legs'][0]['distance']['text'])
print(directions_result[0]['legs'][0]['duration']['text'])
```

Chapter 6

Twitter Module

The usage of the twitter module is explained in this, along with the search parameters that are important for us. Any further user of this project can go through this module and understand the exact implementation of Twitter API.

Standard search API
Returns a collection of relevant Tweets matching a specified query. Please note that Twitters search service and, by extension, the Search API is not meant to be an exhaustive source of Tweets. Not all Tweets will be indexed or made available via the search interface.

Resource URL
https://api.twitter.com/1.1/search/tweets.json

Things / Parameters that we will be using:

Search Word

This will help us sort the tweets based on the words that the tweet contains. We have a dictionary list ready in place, which contains all the words related to traffic jam and its basic causes. So the search result will be a combination of the search result of all these words.

GeoLocation

The location from which the tweet is being made acts as a basic verification step to check whether the user is actually on the road that we want or is he just tweeting stuff from some other place.

Time of the tweet

The time of the tweet is the most important factor because old and obsolete tweets wont give much information about the latest traffic, given that the traffic will always be dynamic and instable.

Tweet Count

This will help us rank the severity of the event. If the number of tweets are more than a given

Parameters Used

Sl No	Name	Required	Description
1	q	Required	URL-encoded search query of 500 characters maximum
2	Geocode	Optional	Returns tweets by users located within a given radius
3	lang	Optional	Restricts tweets to the given language
4	locale	Optional	Specify the language of the query you are sending
5	R-Type	Optional	Specifies what type of search results you would receive.
6	Count	Optional	The number of tweets to return per page
7	Until	Optional	Returns tweets created before the given date
8	Since-ID	Optional	Returns results with an ID greater than the specified ID.
9	Max-ID	Optional	Returns results with an ID less/equal the specified ID.
10	I-Entity	Optional	The entities node will not be included when set to false.

Search Algorithm for tweets:

```
#!/usr/bin/python

import json
import sys
import urllib.request
import os

usage = """
Usage: ./tweet_search.py 'keyword'
e.g ./tweet_search.py accidentatBhandra

Use "+" to replace whitespace"
e.g ./tweet_search.py "accident+at+Bhandra"
"""

# Check that the user puts in an argument, else print the
usage variable, then quit.
if len(sys.argv)!=2:
    print (usage)
    sys.exit(0)

# The screen name in Twitter, is the screen name of the user
for whom to return results for.

# Set the screen name to the second argument
screen = sys.argv[1]

# Open the twitter search URL the result will be shown in json
format
url = urllib.request.urlopen("https://api.twitter.com/1.1/sea
ch/tweets.json?q="+screen)

#convert the data and load it into json
```

```
data = json.load(url)

#to print out how many tweets there are
print (len(data), "tweets")

# Start parse the tweets from the result

# Get only text
for tweet in data["results"]:
    print (tweet["text"])

# Get the status and print out the contents
for status in data['results']:
    print ("(%s) %s" % (status["created_at"], status["text"]))
```

This will print all the related status from the required location. One API key has to be approved from the twitter developer's community to run any code which uses twitter API. Our present use case has not been approved by Twitter yet.

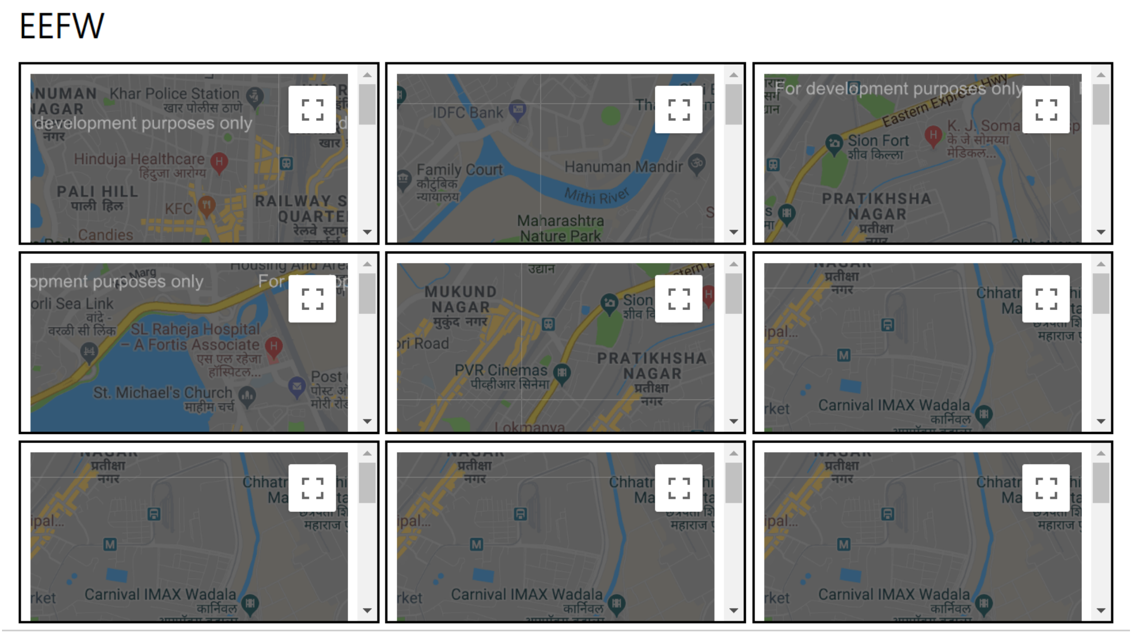
Chapter 7

Webpage for Data Analysis

The final aim of the project is to create a UI/User Interface to display the critical areas where human intervention or traffic police help is required. For the same, I've built a web page which will serve two purposes.

The following are the purposes of the web-page:
1) To display current traffic situation
2) To display graphs obtained from Data Analysis

The current traffic situation is a dynamic display of Google maps of the required location equipped with markers which serve as a purpose to display areas which require attention.



The web-page is built using HTML, CSS and basic JavaScript. It's divided into three different divisions/aspects of the project as follows.

Code Snippet:

```
<div class="w3-third">
    <div class="w3-card w3-container" style="min-height:460px">
    <h3>Coding</h3><br>
    <i class="fa fa-css3 w3-margin-bottom w3-text-theme"
     style="font-size:120px"></i>
    <p>Google Maps API</p>
    <p>Twitter API</p>
    <p>Python 3.6</p>
    <p>HTML and Javascript</p>
    </div>
  </div>

  <div class="w3-third">
    <div class="w3-card w3-container" style="min-height:460px">
    <h3>Analysis</h3><br>
    <i class="fa fa-diamond w3-margin-bottom w3-text-theme"
     style="font-size:120px"></i>
    <p>Data Ingestion</p>
    <p>Plot.ly for Graphs</p>
    <p>Text Recognition</p>
    <p>Display delay</p>
    </div>

  </div>

<div class="w3-third">
  <div class="w3-card w3-container" style="min-height:460px">
  <h3>Webpage</h3><br>
  <i class="fa fa-desktop w3-margin-bottom w3-text-theme"
   style="font-size:120px"></i>
  <p>Display Mpas</p>
```

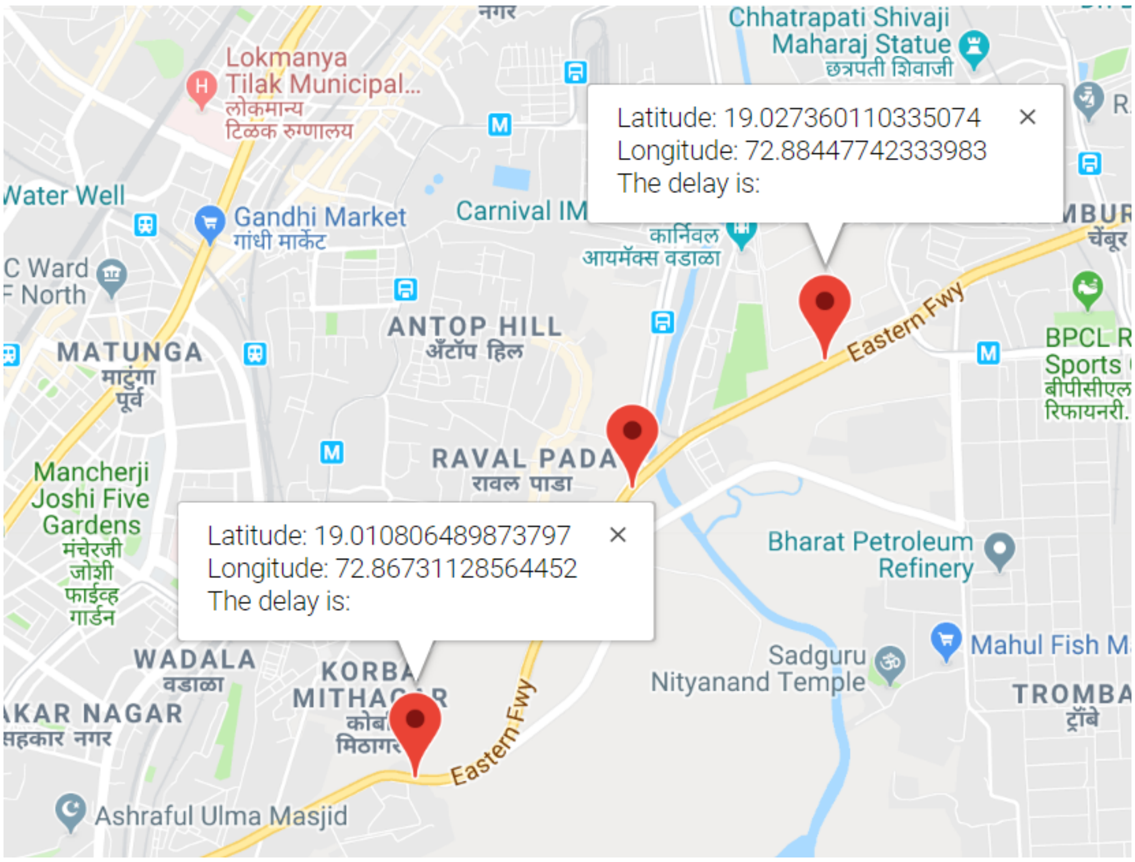
```
<p>Mark Critical Areas</p>
<p>Show the reason for delay</p>
<p>Other Optimum routes</p>
</div>
```

The three aspects of this project are coding, data analysis and building of the web-page. The complete code has been updated on the git-hub repository. Here's the link to the repo: **https://github.com/knotlessguy/BTP**

Chapter 8

Google Maps Marker

This is the most important feature in the UI which will locate the areas which require attention. The marker not only locates the area but also displays the latitude and longitude of that specific area. This is one such feature which can be improved enormously based on the data received through twitter and rank analysis. Here is the image which shows the marker's usage:


Latitude: 19.027360110335074
Longitude: 72.88447742333983
The delay is:

Latitude: 19.010806489873797
Longitude: 72.86731128564452
The delay is:

The following is the logic/code used for the same:

```
<script>
function myMap() {
  var mapCanvas = document.getElementById("map");
  var myCenter=new google.maps.LatLng(19.022735, 72.876581);
  var mapOptions = {center: myCenter, zoom: 14};
  var map = new google.maps.Map(mapCanvas, mapOptions);
  google.maps.event.addListener(map, 'click', function(event) {
    placeMarker(map, event.latLng);
  });
}

function placeMarker(map, location) {
  var marker = new google.maps.Marker({
    position: location,
    map: map
  });
  var infowindow = new google.maps.InfoWindow({
    content: 'Latitude: ' + location.lat() + '<br>Longitude: '
    + location.lng() + '<br>The delay is: '
  });
  infowindow.open(map,marker);
}
</script>
```

Here "Lat" and "Lang" are used as variables. We can also use the time delay as a variable to be displayed in the dialog box which appears with the marker

Chapter 9

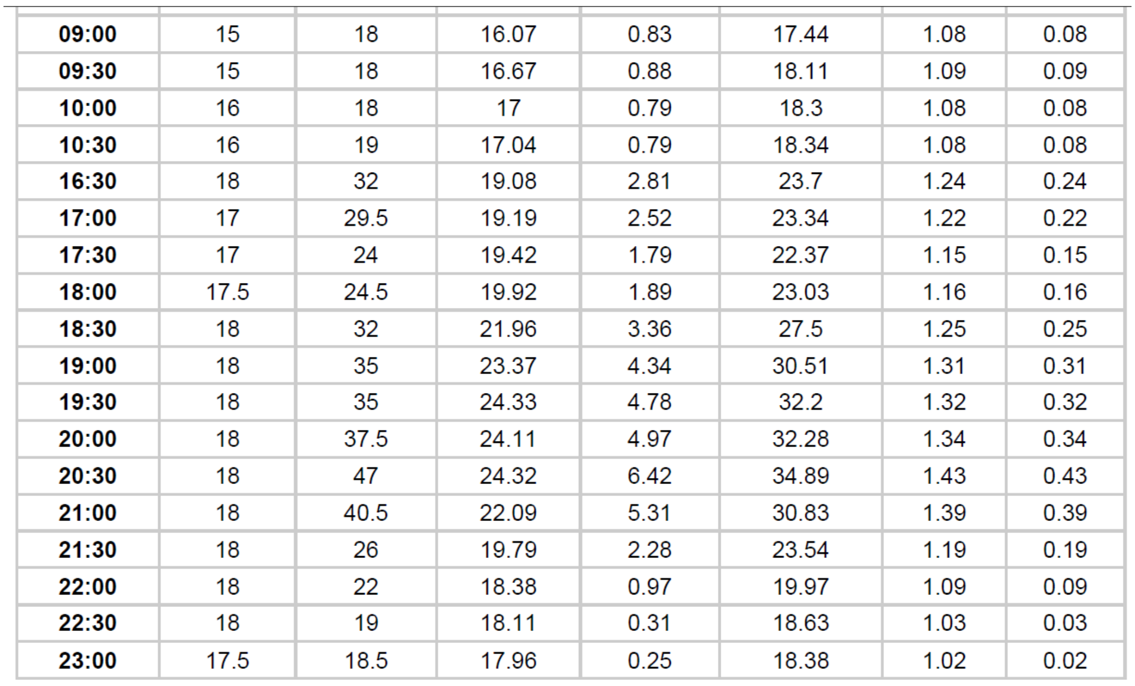
Analysis of data obtained

Data analysis is a process of inspecting, cleansing, transforming, and modeling data with the goal of discovering useful information, informing conclusions, and supporting decision-making. The travel time was estimated using Google Maps API and python script scheduling for various segments of the four highways. The statistical parameters such as minimum, maximum, mean and standard deviation of travel time for the study area at various time stamp in the rush hour are presented in the tables below. Below the table, the variation of BTI is plotted against time of the day for the overall highway.

The data used is quite old as we don't have a Google API key approved yet. Once the key is acquired, new data-set will be obtained by the python code automatically and the data analysis done will refresh based on the new data and their statistical measures. The statistical and reliability measures are estimated from the collected dataset in the rush hour traffic (7:30 to 10:30 16:30 to 23:00) and findings are discussed below:

1.) Eastern Express Freeway (upstream analysis)

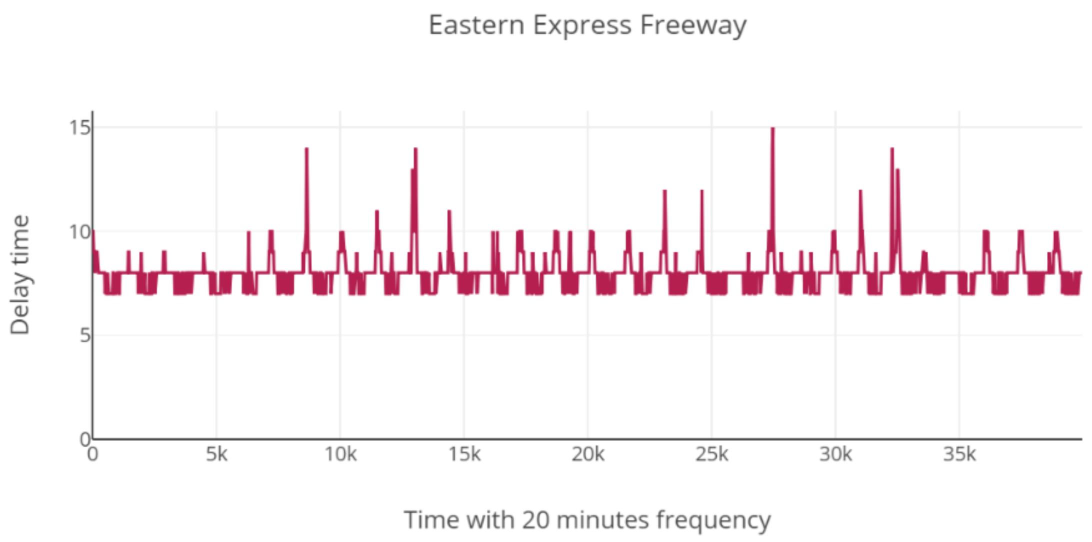
	STATISTICAL PARAMETERS				**RELIABLITY PARAMETERS**		
TIME STAMP	**MIN TIME** *(mins)*	**MAX TIME** *(mins)*	**AVERAGE TIME***(mins)*	**SD.** *(mins)*	**PLANNING TIME** *(mins)*	**PTI**	**BTI**
07:30	15	17	15.68	0.61	16.69	1.06	0.06
08:00	15	17	15.81	0.71	16.97	1.07	0.07
08:30	15	18	16.07	0.72	17.25	1.07	0.07



09:00	15	18	16.07	0.83	17.44	1.08	0.08
09:30	15	18	16.67	0.88	18.11	1.09	0.09
10:00	16	18	17	0.79	18.3	1.08	0.08
10:30	16	19	17.04	0.79	18.34	1.08	0.08
16:30	18	32	19.08	2.81	23.7	1.24	0.24
17:00	17	29.5	19.19	2.52	23.34	1.22	0.22
17:30	17	24	19.42	1.79	22.37	1.15	0.15
18:00	17.5	24.5	19.92	1.89	23.03	1.16	0.16
18:30	18	32	21.96	3.36	27.5	1.25	0.25
19:00	18	35	23.37	4.34	30.51	1.31	0.31
19:30	18	35	24.33	4.78	32.2	1.32	0.32
20:00	18	37.5	24.11	4.97	32.28	1.34	0.34
20:30	18	47	24.32	6.42	34.89	1.43	0.43
21:00	18	40.5	22.09	5.31	30.83	1.39	0.39
21:30	18	26	19.79	2.28	23.54	1.19	0.19
22:00	18	22	18.38	0.97	19.97	1.09	0.09
22:30	18	19	18.11	0.31	18.63	1.03	0.03
23:00	17.5	18.5	17.96	0.25	18.38	1.02	0.02

Table 2 Statistical and reliability analysis of EEF(upstream) in rush hours

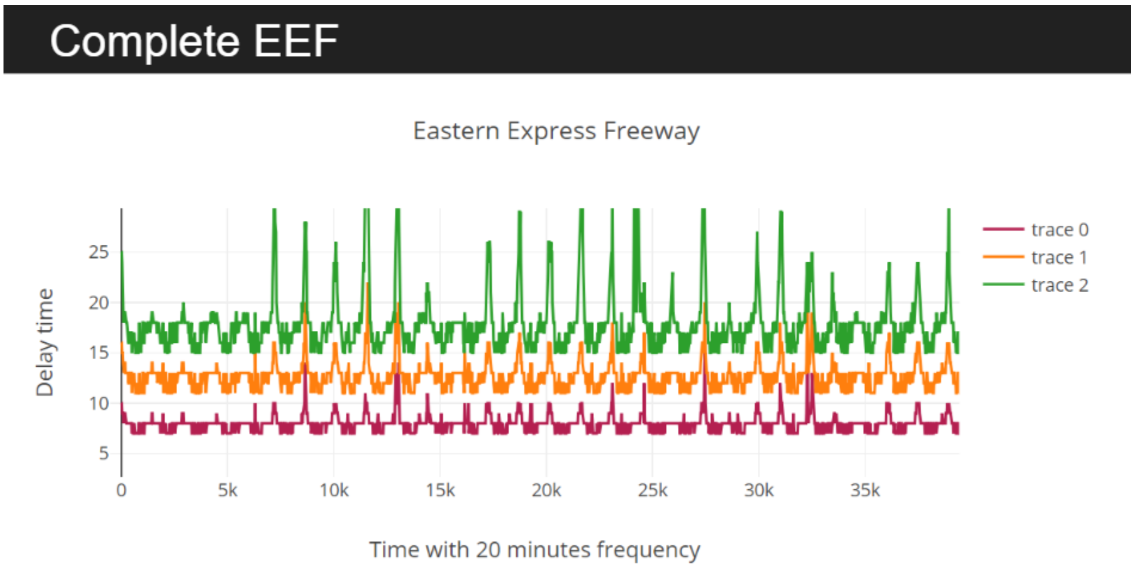
Here is the link to a short presentation made on all the data available:
http://bit.ly/Data-Analysis-BTP
There are different parameters which can be used to analyse the current situation, one is the BTI and the other one is the time delay. Presenting both the graphs here:



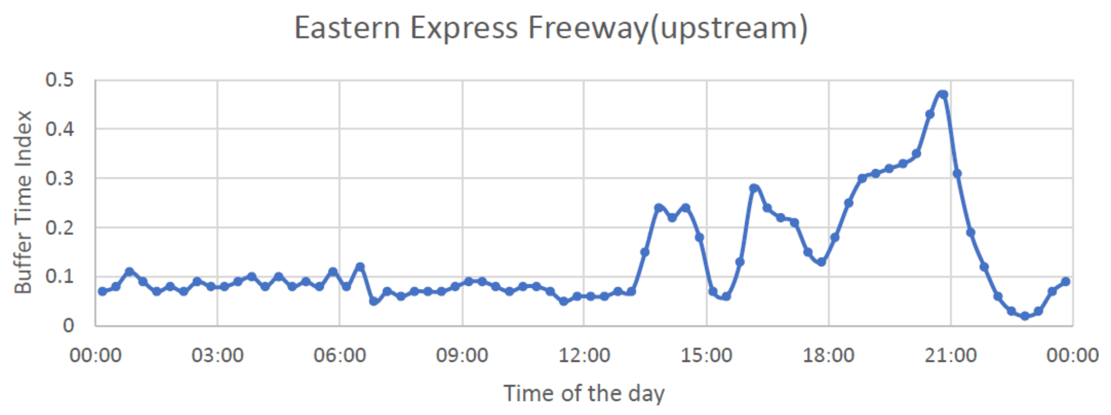
Observed Trends
1) We can clearly observe a base amount of delay that always stays.

2) We can fix this as our minimum delay and set the buffer amount of delay to 2-3 minutes based on the data.
3) This is just for the patch A-B Global maximums depict serious/severe events
4) As we can observe many local minimums in the graph this shows the need for a proper validation method of our events

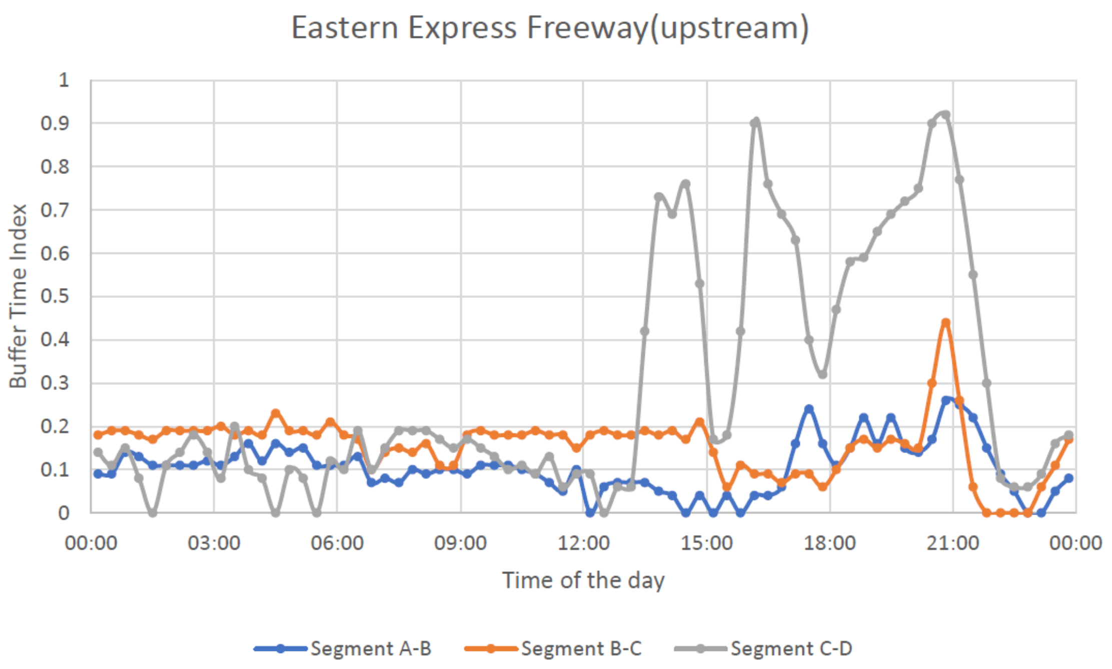
The following is obtained when the complete EEF is taken into consideration:



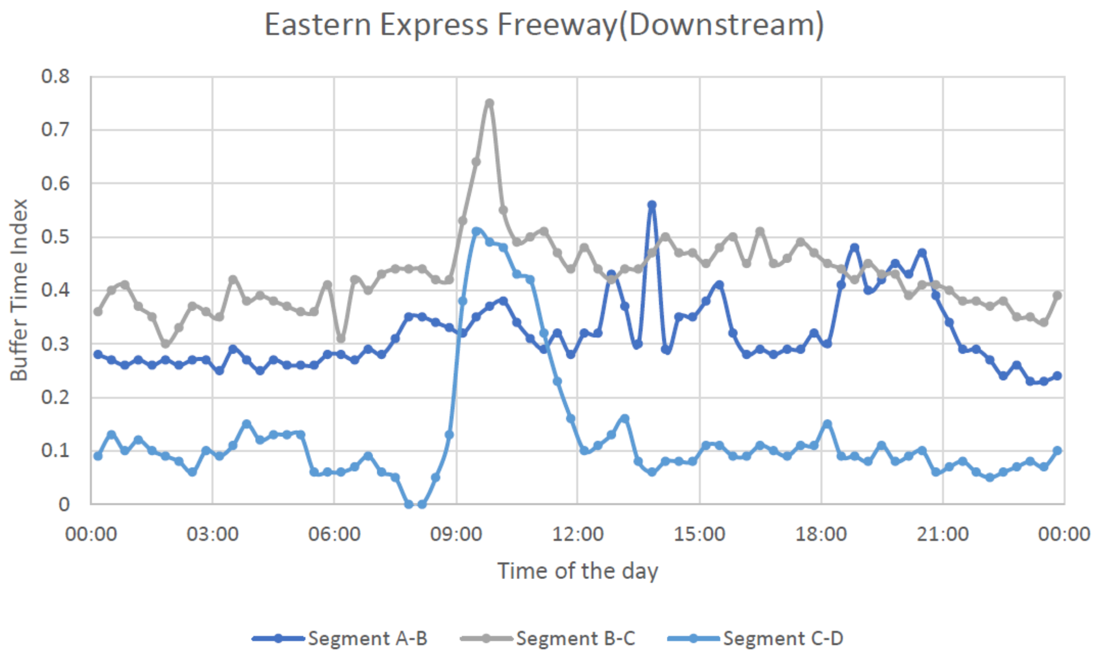
This is the graph obtained using **Buffer Time index** as a parameter.



By analyzing the segmental graph we can see that the Segment C-D has the highest BTI as compared to other two segments(which are fairly consistent). By comparing the BTI of EEF with other highways, we can see that the Eastern Express Freeway is the most reliable. Here is the segment-wise analysis:



Analysis for other freeways and other streams are done on the same parameters, attaching all their graphs here:



From the graph it is clearly visible that downstream is more consistent than the upstream. These are the results which are desired from data analysis while deciding the most reliable paths

Chapter 10

Conclusion and Scope of Improvement

Travel time reliability are very important in estimating the operational efficiency of a roadway. Travel time has been collected through Google Maps API by using python scripts for 4 major roadways of Mumbai. This study identified the requirement of travel time reliability measurements for measuring performance of roadways of Mumbai with Statistical and Travel time reliability analysis.

The problem intended to solve is quite large and a lot of progress is done in that direction. There are a lot of things that have to be done. Listing them down on the basis of their priority:
1) Google Maps API key has to be obtained
2) Twitter API key has to be obtained
3) A proper Django framework has to be scripted to implement backend database
4) The amount of data collected increases day by day. The old data has to be flushed accordingly by generalising an offset.
5) Updating of the maps should be done dynamically, as this project has to serve in current runtime not in future.
6) The algorithms used has to be optimized to make sure that the backend runtime does not cross a certain limit.