SAS® GLOBAL FORUM 2018

USERS PROGRAM

Reducing Traveling Times for the Cobb County Fire Department

April 8 – 11 | Denver, CO

#SASGF

Reducing Traveling Times for the Cobb County Fire Department

Bogdan Gadidov, Lili Zhang, Yiyun Zhou, Ph.D. Students in Analytics and Data Science

Advising Faculty: Dr. Joe DeMaio, Dr. Kurt Schulzke

Kennesaw State University

Abstract

Cobb County is a suburban county located northwest of Atlanta. The Cobb County Fire Department's (CCFD) 8-minute emergency response time doubles the National Fire Protection Association's (NFPA) 4-minute standard, measured at the 90th percentile of all emergencies. This project aims to reduce CCFD's response time by focusing on the travel times of their various emergency vehicles. Currently, there are 29 fire stations and 272 fire zones within Cobb County, with each fire station being responsible for a pre-defined set of fire zones. We investigate whether fire zones and stations can be realigned to reduce travel times by analyzing historical response time data from September 2011 to August 2016. CCFD historical data reveal which fire station actually responded to each incident, as well as the related travel time. Google Maps is then used to check the response times from neighboring fire stations to determine whether a different fire station could have responded more quickly to the same incident. The comparison between historical and Google travel times reveals the location and frequency of disagreement between the historical and Google recommended fire stations. Fire zones can then be reassigned to different fire stations to reduce future traveling times. Results vary for each fire station, but they show that there is room for improvement in the way that CCFD currently responds to emergencies. Python is used to connect to the Google Maps Distance Matrix API, and SAS® is used for resulting analyses.

Methods

To perform this analysis, Google Maps is used to take into account the geography, roadways and traffic congestion of Cobb County as it is today. Google Maps is used to scan historical incidents to find cases where a different fire station than the historically used station could lead to faster response times. Python is used to connect to Google Maps and calculate traveling times in real time. For any historical incident, the Google time is first checked for the corresponding station. Then neighboring fire stations are also checked to see if any other stations could respond faster than the corresponding fire station to the incident. The comparisons of these times are used to drive recommendations on which zones should be reassigned to overall decrease traveling times. Additionally, Cobb County could subdivide fire zones and assign the resulting pieces to different stations. A list of backup fire stations for each fire zone is also created using the Google Maps simulation since a distribution of the fastest responding station is created for all incidents in each station's current area. Results vary for each fire station, but they show that there is room for improvement in the way that CCFD currently responds to emergencies. The Google Maps Distance Matrix API is accessed to retrieve the real time traveling times. Figure 1 on the right shows the current map of Cobb County with the 29 different colored regions which represent the area which each of the 29 fire stations is responsible for. Figure 2 shows the graph of the Cobb County fire stations with the neighboring stations which was used to determine which station could respond in the shortest amount of time to each incident.

Methods (cont.)

Figure 1: Map of Current Cobb County Fire Zones and Stations

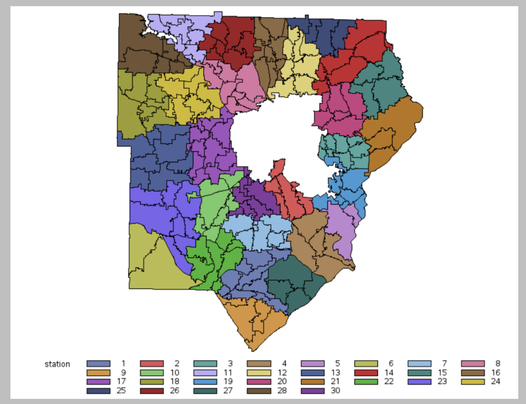
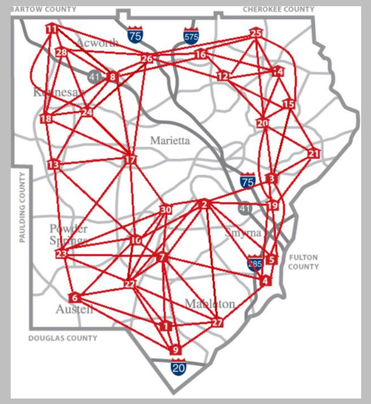


Figure 2: Graph of Fire Stations With Neighboring Stations



The steps in creating the recommendations is summarized below:

1. Retrieve relevant information of all incidents (address, roll time, station number) from the historical data
2. Retrieve the addresses of all 29 Cobb County fire stations
3. Create a correspondence table to be used for each incident. Each fire zone is given a list of its neighboring fire stations (Figure 2), to be used to determine which fire station can respond most quickly.
4. Using the Google Maps Distance Matrix API, calculate the traveling time from each historical incident to its corresponding neighboring stations. Select the station which can reach the incident in the shortest amount of time.
5. Compare the percentage of the time that the Google Maps recommended station matches the historically used station. In fire zones where the plurality of Google Maps recommendations are different than the historically used station, re-assign this zone to a new station. Tables 1 and 2 on the next slide show an example for several zones which are candidates for reassignment.

Reducing Traveling Times for the Cobb County Fire Department

Bogdan Gadidov, Lili Zhang, Yiyun Zhou, Ph.D. Students in Analytics and Data Science

Advising Faculty: Dr. Joe DeMaio, Dr. Kurt Schulzke

Kennesaw State University

Results

Table 1: Recommendations for Zones 17H, 17J and 17K

Zone	Google Recommended Station		
	8	10	17
17H	1%	74%	25%
17J*	3%	40%	56%
17K	1%	82%	16%

Table 2: Recommendations for Zones 8H, 8I

Zone	Google Recommended Station		
	8	17	24
8H	10%	88%	1%
8I	39%	20%	41%

It is recommended that 12 zones be reassigned to different fire stations. In these zones, there a positive reduction in the average response time by reassigning incidents in these zones to a different fire station. Figure 3 shows the updated map of Cobb County with the changes. This map can be compared to Figure 1 to see the differences in zone reassignments. Table 3 shows the 12 zones and their newly recommended station assignments. The average time savings is also reported in the table. This time savings is again the difference between the Google time when using the assigned station to using the "best" station. The time savings are reported by time of day, as the traveling times for the incidents were gathered at the same time of day that the incident was reported at. Recommendations are largely the same across the time of day, but some zones have varying time savings at different times of the day.

Figure 3: Map of Cobb County Fire Zones with Recommended Reassignments

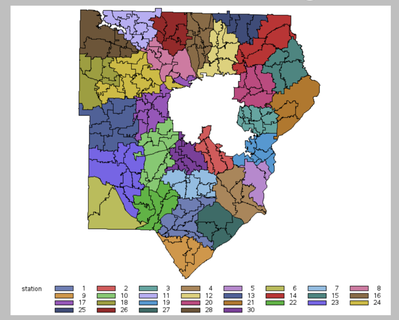


Table 3: Summary of Average Time Savings for Each Reassigned Zone

		Morning		Midday		Evening		Night	
Zone	Recommended Station Assignment	Number of Incidents	Time Savings	Number of Incidents	Time Savings	Number of Incidents	Time Savings	Number of Incidents	Time Savings
1C	27	113	3	330	3	253	2	321	3
7A	30	10	166	43	169	34	219	31	231
8H	17	128	79	346	69	232	70	310	58
8I	24	65	63	188	45	158	49	156	24
17H	10	59	30	99	30	91	31	126	33
17J	10*	58	17	127	8	72	1	113	.
17K	10	54	68	136	80	88	12	134	25
18A	24	58	63	148	73	66	60	123	82
18F	24	33	27	75	22	67	1	87	14
19F	3	180	84	365	85	289	84	450	83
26D	8	90	30	204	20	135	19	204	22
26G	11	74	25	178	10	149	13	190	11

Relevant Code

Figure 4: Select SAS Code

```
* import shape file which defines boundaries of cobb county;
proc mapimport datafile='/gpfs/user_home/FireZones_8_31_20161.shp' out=allzones;
run;

* create map of cobb county;
proc gmap data=zones map=zones;
id objectid;
choro station /discrete;
run;

* compute reduction in travel times with newly recommended zones;
proc means data=mid;
var corr_station_google_time google_time;
where zone='8I' and google_station=24;run;

* code to create Table 1;
proc freq data=mid ;tables zone*google_station / nocol nopercent;
where zone in ('8A','8B','8C','8D','8E','8F','8G','8H','8I');run;
```

Figure 5: Select Python Code

```
origins = []
for station in stations_to_check:
    address = fire_facilities['ADDRESS'][fire_facilities['STATION NUMBER']==station].values[0]
    origins.append(address)

try:
    directions_result = gmaps.distance_matrix(origins, [destination])
except:
    raise

traveling_time = []
for row in directions_result["rows"]:
    if "duration" in row["elements"][0].keys():
        traveling_time.append(row["elements"][0]["duration"]["value"])
    else:
        traveling_time.append(0)

station_time_pairs = {}
for i in range(len(stations_to_check)):
    station_time_pairs[stations_to_check[i]] = traveling_time[i]
sorted_station_time_pairs = sorted(station_time_pairs.items(), key=operator.itemgetter(1))

station_number = sorted_station_time_pairs[0][0]
google_stations.append(station_number)

min_time = sorted_station_time_pairs[0][1]
google_time.append(min_time)

print "Best Station (station, time in seconds): "
print "(%s, %s)" % ( station_number, min_time)
```

SAS® GLOBAL FORUM 2018

April 8 – 11 | Denver, CO
Colorado Convention Center

#SASGF