[DRAFT]
On-demand public transport is making us mobile

RA

2021-06-19

1 Introduction

The TLC trip record data[1] records Yellow and Green taxi and other "for-hire vehicle" trips in New York City. In the earlier years in particular, Yellow and Green taxi records contain timestamps, trip distance, passenger count, fare, etc., but also approximate locations of pickup and dropoff. Based on those data we give a partial answer to the question

> How many small buses could serve the same demand?

Conventions. The number in the margin refers to the corresponding code listed in §6.1.

2 Data preparation

2.1 Taxi trips[2]

We focus henceforth on May 2016 where ∼12M (Yellow) and ∼1.5M (Green) trip records are available. The ∼11M "for-hire vehicle" records bear no useful details for our purpose. We keep only the trips that begin and end in Manhattan with reported trip distance between 0.1 and 30 miles. We filter out records that lack geo-coordinates. See Fig. 1 for a net summary. #1

2.2 Road graph[2]

We obtained the road network for Manhattan from the OpenStreetMap Overpass API[3] and #3
filtered for roads that can plausibly sustain public traffic. It is represented as a digraph, #4
i.e. there are one-way roads and the routing $A \to B$ differs from $B \to A$. When modeling

[1] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[2] The codes for this section were mostly written in 2019.
[3] https://wiki.openstreetmap.org/wiki/Overpass_API

individual trips, their reported pickup and dropoff locations are snapped to the nearest node of the road graph; we ignore about 10% of the trips where the discrepancy is over 20 m.

The map graphics are from MapBox. #5

2.3 Traffic model

The trip trajectories are not available, only the pickup and dropoff locations (with potential GPS uncertainty of 10 to 100 m). We leverage the reported trip duration to infer plausible mean-field travel velocities (see Fig. 2c) throughout the road graph iteratively as follows. The velocities on all roads are initialized to 5 m/s. For a sample of trips that start at 18-19h the quickest trajectories are estimated. The velocities of the road bits participating in those trajectories are adjusted toward the reported trip duration. This is repeated. #6

The adaptive routes we calculate below are the quickest trajectories w.r.t. this traffic model.

3 Optimization problem

We are facing a so-called vehicle routing problem with these main attributes:

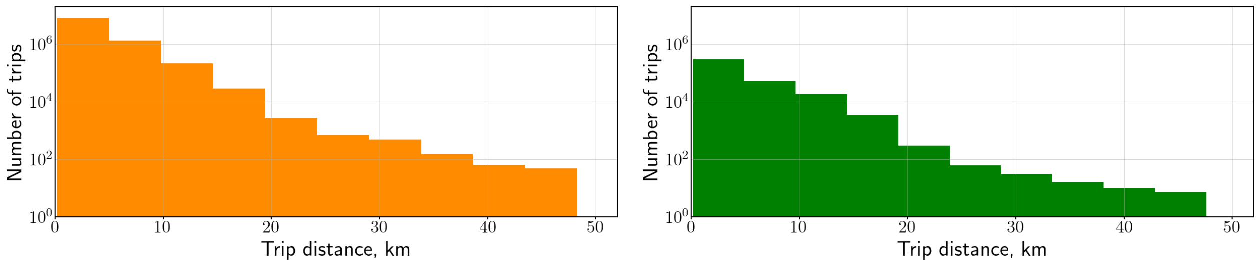
- Capacitated. There are N buses of maximal capacity of C passengers each.
- Time windows. Each passenger has to be picked up within $[-2\,\mathrm{min}, 5\,\mathrm{min}]$ of the recorded pickup time in the trip data (§2.1). The dropoff time window extends to 10 min after the recorded dropoff. To ensure feasibility, a passenger may be ignored at a certain penalty to the optimization objective. The buses are allowed to wait up to 10 min at any location.
- Depot. All buses start and finish at a certain location but have enough time to reach anywhere without compromising feasibility.

This is difficult in general. We use `ortools`[4] to find reasonable solutions computationally. We can roughly assess optimality by allotting more time to the solver.

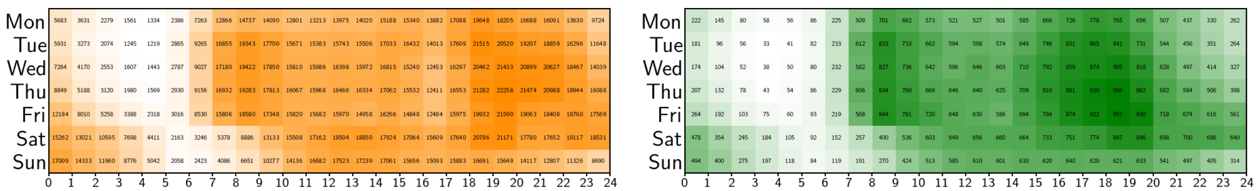
To obtain reasonable solutions on modest hardware we focus on a small slice of the trip data at a time, i.e. a few hundred passengers × one hour × a few square kilometers. We then compare "customer satisfaction" across different fleet sizes N and bus capacities C.

We pretend here that the demand and the traffic conditions are known in advance, whereas some 10 min in advance would be more realistic. Meanwhile, the density of requests in space and time is quite high. Thus, we believe our results remain informative.

[4] https://developers.google.com/optimization/routing/vrp

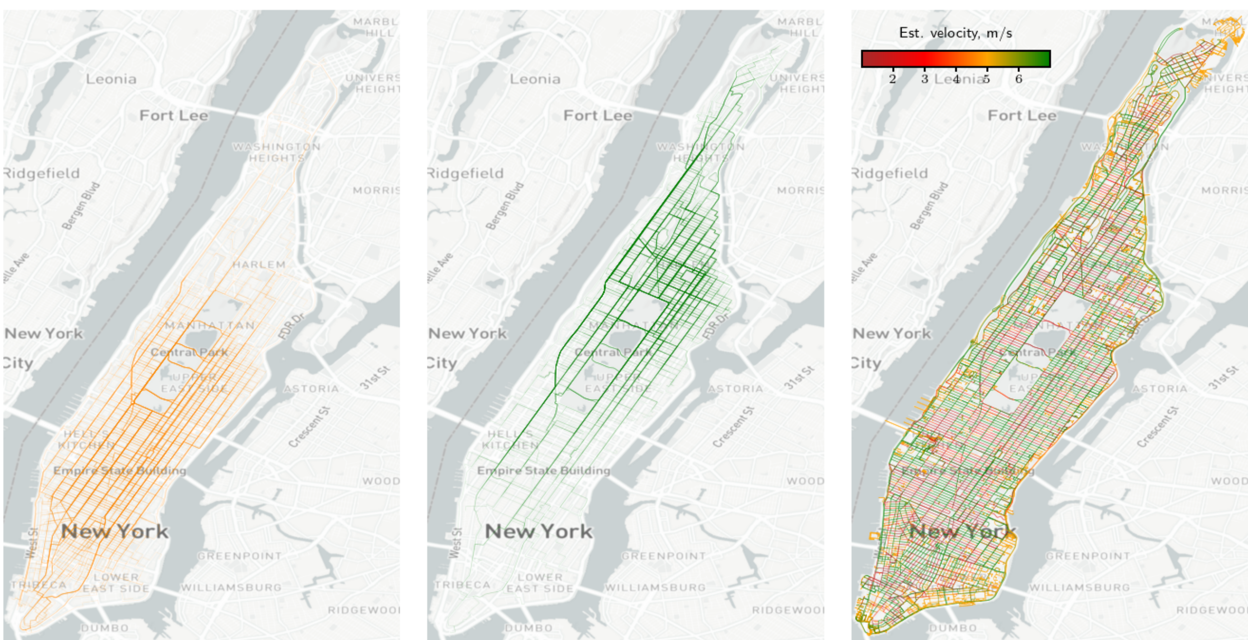


(a) Reported trip distance histogram.



(b) Pickup hour heatmap.

Figure 1: Summary of Yellow (↖) and Green (↗) taxi trips filtered as in §2.1. #2



(a) Sample Yellow taxi trips. (b) Sample Green taxi trips. (c) Inferred mean-field velocity.

Figure 2: A sample of shortest-path trajectories (§2.1/§2.2) and the traffic model from §2.3. #7, #8

4 Case study

4.1 Times Square

We take the first n single-passenger trips with reported pickup and dropoff within $R = 1\,\mathrm{km}$ #9
of Times Square and within 18:00–19:00 on May 1, 2016. Allowing $n = 400$ requests is a little difficult to solve ($\sim$1 h $\times$ 2 Gflop/s), so we focus here on

$$n = 100 \text{ requests for } N = 10 \text{ vehicles of capacity } C = 1 \text{ or } C = 8\ .$$

We find that the single-passenger fleet ($C = 1$) can only service about a half of requests, whilst the fleet of minibuses ($C = 8$) can handle most requests, see Fig. 4.

The vehicle load over time is shown in Fig. 5, indicating that it runs at about half the capacity. Since all $n = 100$ pickup requests are from the first 17 min only, the fleet would be able to handle more requests per minute if operating over an extended period.

TODO(1): more on this?

TODO(2): another case study?

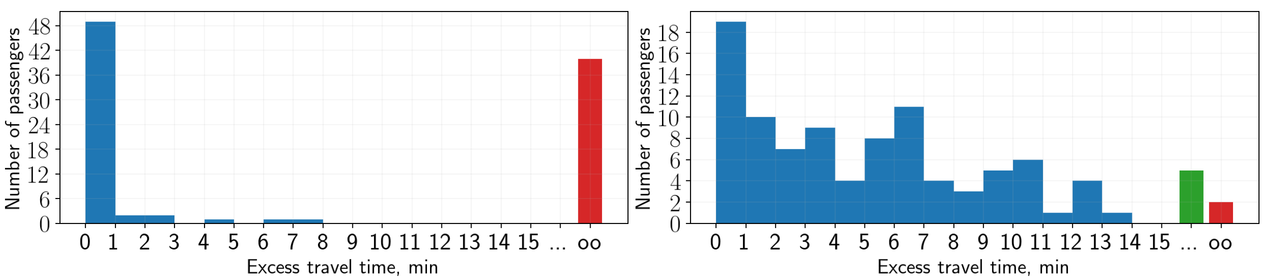
5 Conclusions

TODO(3): drop the mic

(a) Vehicle capacity $C = 1$.

(b) Vehicle capacity $C = 8$.

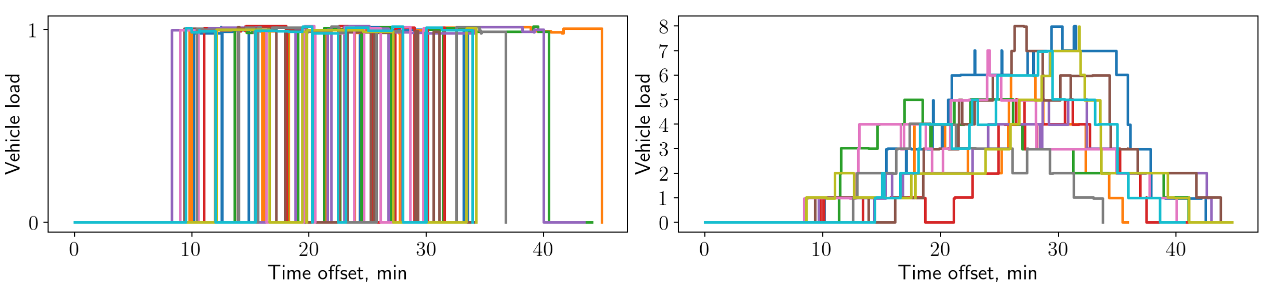
Figure 3: Passenger trip trajectories colored by excess travel time w.r.t. quickest route ($n = 100$ requests for $N = 10$ vehicles). Empty red circles are unserviced requests. See §4.1.



(a) Vehicle capacity $C = 1$.

(b) Vehicle capacity $C = 8$.

Figure 4: Histogram of excess travel time w.r.t. quickest route ($n = 100$ requests for $N = 10$ vehicles). See §4.1.



(a) Vehicle capacity $C = 1$.

(b) Vehicle capacity $C = 8$.

Figure 5: Load of each vehicle over time ($n = 100$ requests for $N = 10$ vehicles). See §4.1.

6 Appendix

6.1 List of codes

	page	https://github.com/numpde/optimum/blob/main/ …
#1	p.1	code/data/20210610-NYCTLC/a_download.py
#2	p.3	code/data/20210610-NYCTLC/e_explore.py
#3	p.1	code/data/20210611-OSM/a_osm_download.py
#4	p.1	code/data/20210611-OSM/c_road_graph.py
#5	p.2	code/helpers/opt_maps/maps.py
#6	p.2	code/model/20210613-GraphWithLag/b_train.py
#7	p.3	code/data/20210611-OSM/e_explore.py
#8	p.3	code/model/20210613-GraphWithLag/b_train.py
#9	p.4	code/work/20210616-OPT1

6.2 TODOs

1. p.4. more on this?
2. p.4. another case study?
3. p.4. drop the mic

2021-06-19 15:03:39+02:00