Alexport: Design and Analysis on a New Form of Transportation Network

Hollis Liu
School of Computing
Clemson University
Clemson, SC, USA
hanjiel@clemson.edu

Dr. Ilya Safro
School of Computing
Clemson University
Clemson, SC, USA
isafro@clemson.edu

***Abstract*—Overall this paper is about the design and evaluation of a new kind of traffic network defined by a premise: automobiles are separated into two parts: The engine and the interior. There comes many benefits and ramifications with this new form of transportation that needs investigation. The author names this new form of transportation the Alexport and the key ideas around it are explained in detail.**

***Index Terms*—transportation, network, graph, simulation**

I. Introduction

With the rise of autonomous driving, numerous innovation opportunities will emerge in the field of transportation. The first author proposes a modular modification to existing cars, namely, a new personal transportation system where automobiles are separated into two parts: The engine and the interior. The **engine** part includes all the mechanical components and energy storage needed for the vehicle to self drive. That includes: battery cells, motors, suspension, brakes and so on. The **interior** includes all the things that makes a car personal and comfortable: The frame, the console, the seats, HVAC, speakers, etc. There also should be mechanisms on either side of the car components to enable quick and easy detach and docking, as well as reliable suspension.

This new kind of transportation system is named Alexport by the author for referencing ease. The high level abstraction of Alexport can be described as a bipartite graph with interior nodes in one group and engines nodes in the other. The edges connecting them will be undirected edges with weights associated with them. The weight is usually the geographical distance between the nodes. This paper mainly focuses on the analysis of the aforementioned network.

II. Motivation

The logic behind this rather radical design roots from the observation of sharing economy, especially in ride sharing. It has become more and more realistic that private cars can be shared autonomously. However, sharing a private car can be problematic:

A. Privacy and Individualism

People still love a sense of private ownership and individuality of their cars. If all cars are shared ones, then there is no room for personalized design choices nor will the car be a personal space that is tidy and hygienic.

B. Cost

Someone still has to own a car. The cost of owning a car is very high because of the manufacturing and maintenance cost of the engine. If we can eliminate the cost of owning an engine while keep the ownership of the car interior, a car can be a lot more affordable.

C. Interfacing

Separation of the interior can also enable seamless and future-proof interfacing between local transportation and higher speed global transportation systems such as the high speed rails or Hyperloop [1].

III. Design and Contribution

Since Alexport is an abstraction of a real world application, it only makes sense to design and test Alexport based on a real world scenario. Thus the design process is broken down into four parts: data collection from real world maps, define and assign graph components, define constraints and simulation of the network.

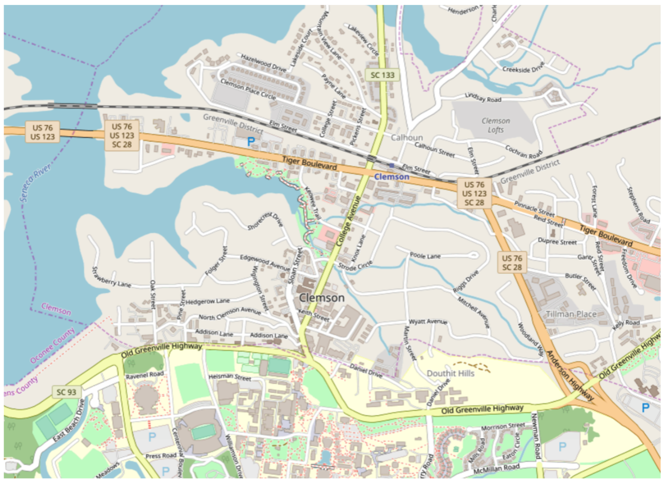

Fig. 1. Map data used for the Alexport simulation mentioned in this paper

A. Data Collection

In order to gather real world data that is authentic and representative, the author downloaded and parsed a region that encompasses part of Clemson University, downtown Clemson and its adjacent neighborhoods. The captured area is showed below in figure 1. The map data is from OpenStreetMap [2].

The network analysis on Alexport is highly scenario based as different travel pattern is presumed to yield drastically different dynamics in the network. The first scenario in discussion is the morning commute.

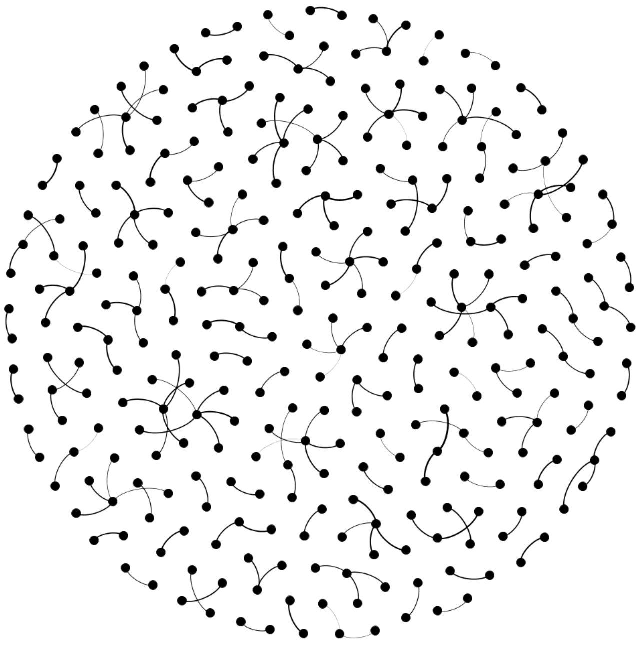

Fig. 2. Visualization of a morning commute in Alexport

Since OpenStreetMap provides the map data in an open format, the author was able to export this exact map into a xml format. A Python script is written to parses the xml file, filter out the region of interest (apartments and school buildings), compute distance between ROIs and compile all of them into a graph representation using NetworkX [3]. The Python script is included as ”parse_map.py”.

As evidenced by the execution of program, there are 207 apartments and 125 universities buildings in this region shown in figure 1. Thus one university building node can have multiple inward edges but one apartment node can only have one outgoing edge. The weight of the edge is defined as the physical (Euclidian) distance between two nodes computed based on the latitude and longitude data.

A visualization of this particular graph is shown in figure 2. Fruchterman Reingold [4] layout in Gephi [5] was used to visualize this graph. The edges stand for all possible commuting routes and the thickness of the edge represents the physical distance between nodes. The specific routes are randomly assigned.

A second scenario simulates a surging event. During a surging event, everyone in the area wants to go to a singular place such as a football game or a concert venue. A graph generated using the same visualization technique is included as figure 3.

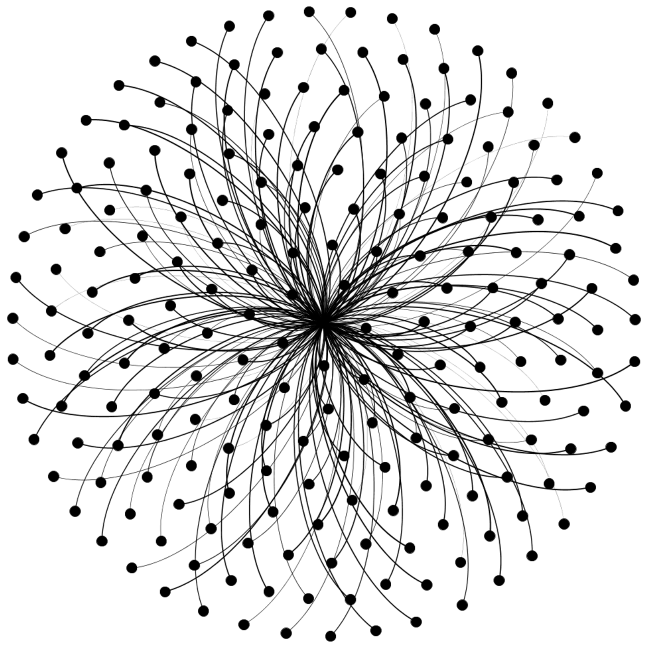

Fig. 3. Visualization of a surging event in Alexport

B. Graph Definition

In figure 2 and 3, the nodes represent the users while the edges represent a commute path with its thickness signifying edge weight based distance. It is somewhat naive to assign euclidian distance as the commute distance. But considering there is usually a linear relationship between the actual commute distance and euclidian distance and for the sake of simplicity, euclidian distance is unapologetically used in this paper to represent the edge weight.

What is missing from figure 2 and 3 is the visualization of interior nodes and engine nodes. They naturally form subgraphs of the whole graph since the amount of interior or engine nodes will never exceed the amount of apartments. The exact amount of those nodes and the ratio between them are one of the key observations of this network. It is also noted that the amount of interior nodes will be kept the same as the amount of users. The logic behind this decision is to ignore the case where one house hold consisting of multiple members might only own one interior.

The total execution time is the key observation of Alexport and the main minimization goal of this paper. The total execution time (TET) is defined to be the time duration for all users in a region to be transported to their desired destinations. Note that there might not be an engine node at an interior node at a given time so an engine node needs to travel extra miles just to get to an interior node, adding the extra travel time and thus increasing TET.

C. Constraints

It is not difficult to realize that the more engine nodes there are in the system, the lower the TET (total execution time). However, the whole purpose of Alexport is to reduce the amount of engines so the overall cost of a transportation system can be lowered. Another problem rises if there are too few engine nodes: the TET will dramatically increase if most users need to wait for an engine node to come. Thus balancing the ratio between interiors and engine nodes is crucial to the viability of Alexport.

Another optimization constraint is the distance of engine nodes from a user initiating a ride request. Alexport needs to pick out the closest engine node from all the available engines to minimize wait time and TET.

D. Simulation

With Alexport setup for the 2 mentioned above scenarios, simulation is a good method of investigating the dynamics of the graph and determining the most fitting ratio between interior and engine nodes. The included file ?simulate.py? handles the simulation task. This object oriented program models requests, users, engines and rides in multiple states. Necessary details such as engine speed and extra travel distances are all taken into account. The code in its entirety is included in the source file ”simulate.py”.

The main simulation loop with time step set to 1 minute encloses all the interactions within Alexport such as initiating a request, finding the nearest engine node and finishing a ride. Listing 1 below shows the code snippet that is in charge of the interactions for each time step in the Alexport simulation. The function eventually returns the number of minutes that corresponds to TET for a scenario.

```
clock = 0  # unit in minute
time_step = 1 # minue
while active_ride and requests:
    for index, a in enumerate(active_ride):
        a.distance -= a.engine.speed * time_step

        # a ride is complete
        if a.distance <= 0:
            active_ride.pop(index)
            available_engine.append(Engine(a.engine.
    tag, a.dest_lat, a.dest_lon))

    # assign new rides
    if available_engine:
        curr_request = requests.pop()
        curr_request.engine = available_engine.pop(
    find_nearest_engine_index(available_engine,
    curr_request.origin))

        # account for the initial wait
        curr_request.distance += distance(
    curr_request.engine.location, curr_request.
    origin)
        active_ride.append(curr_request)

    clock += time_step
```

Listing 1. main simulation loop

IV. Results

Total execution time (TET) is recorded for the two aforementioned scenarios based on different interior to engine node ratio. Figure 4 through 7 show the total execution time for a certain scenario plotted against the interior node to engine node ratio. The ratio is calculated as the amount of interior nodes divided by engine nodes.

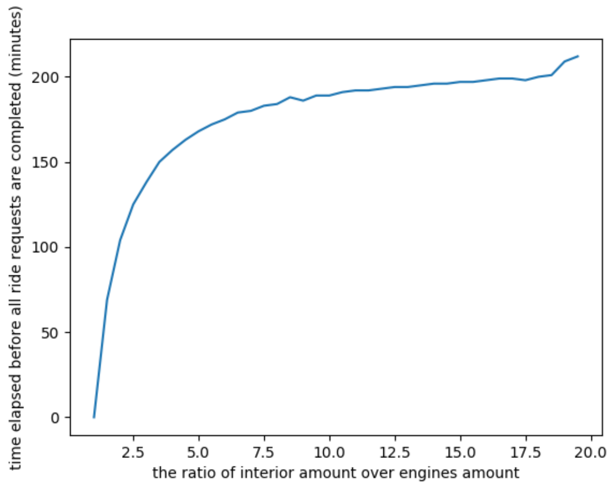

Fig. 4. Morning commute TET based on interior/engine ratio from 1 to 20

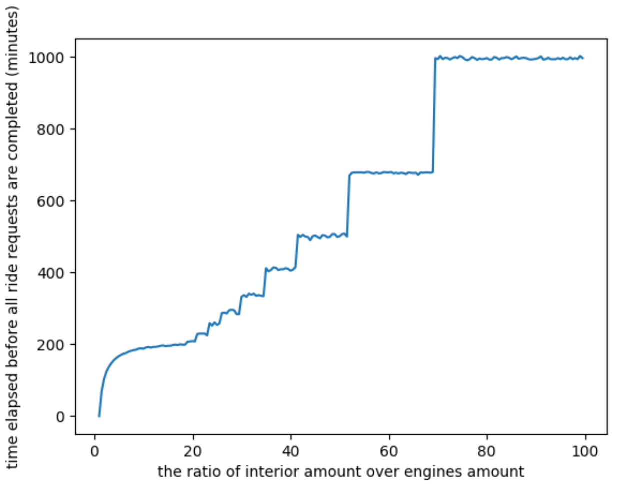

Fig. 5. Morning commute TET based on interior/engine ratio from 1 to 100

The above two figures show the same scenario at two different scales. In the first scenario of morning commute, it is easy to observe that before the ratio reaches 20, TET grows mostly logarithmically and plateaued after around 7.5. The TET starts to grow in dramatic fashions after reaching the ratio of 20 and grows in intervals. The TET also fluctuates within each observed interval.

The results are mostly similar for the surging event scenario but with less fluctuations as evidenced in figure 6 and 7.

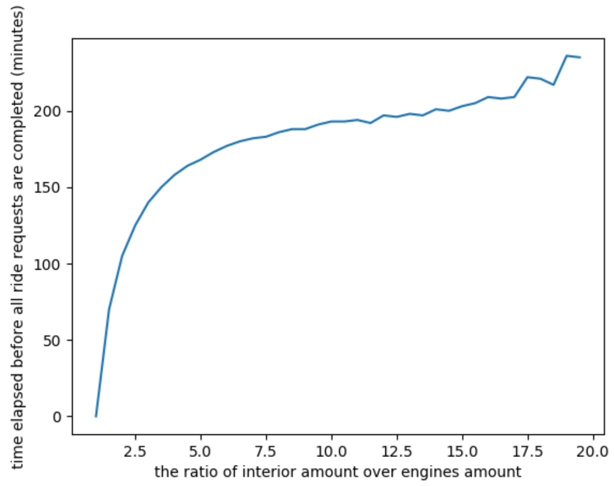

Fig. 6. Surging event TET based on interior/engine ratio from 1 to 20

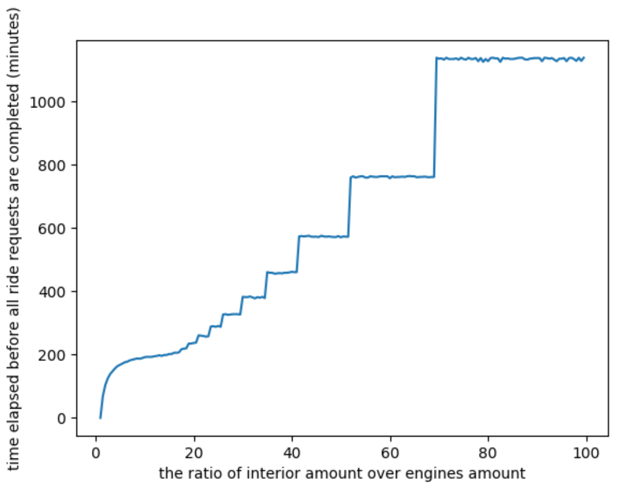

Fig. 7. Surging event TET based on interior/engine ratio from 1 to 100

V. Conclusion

The result of total execution time is certainly surprising given the drastically different behavior for different interior/engine ratio and also the similarity for two dramatically different scenarios.

Generally speaking, if the budget is reasonable for Alexport, to aim for an interior/engine ratio of 7-10 is a good estimate. This is to say for every 100 users/interiors, 10 - 14 engines need to be deployed to reach time-cost balance. But if the budge is extra tight, then the city planner need to look at the intervals range to make the best decision as a ratio of 80 is not at all different from the TET result of ratio 100.

A conclusion can also be drawn that there is no need for special planing in Alexport for surging event as the simulation shows little to no different in TET at a given interior/engine ratio for the two scenarios.

VI. Discussion and Related Work

Since Alexport is an original idea, there is no related academic work the author can find. It is however, rather obvious to compare Alexport to existing ride sharing service such as Uber and Lyft so it is worth noting their difference: Alexport is designed on a totally different premise, that is cars can be separated into an interior part and an engine part. What it enables is not only a privacy-first ride sharing experience, but also the freedom of choosing the configurations of engines and minimizing the overall cost of a transportation system.

Dynamic graph was also a topic of interest for Alexport but since the simulation takes into account dynamic amount of engine nodes in an extended range, there is no need for standalone discussion of dynamic graph in this paper.

VII. Future Work

Future work of Alexport includes testing the network on larger datasets, introducing more variables for engines, interiors and users. In addition, collaborative work with the mechanical engineering department and automobile innovation center is also a logical next step. Considering Alexport is merely an idea in its infant stage, the future is full of possibilities.

VIII. acknowledgements

Map data copyrighted OpenStreetMap contributors and available from https://www.openstreetmap.org"

This work is initially for the purpose of a course final project at Clemson University: CPSC 8480 Network Science, taught by Dr. Ilya Safro.

References

[1] Musk, Elon (August 12, 2013). "Hyperloop Alpha" (PDF). SpaceX. Retrieved August 13, 2013.

[2] OpenStreetMap contributors. (2015) Planet dump [Data file from Latest Weekly Planet XML File]. Retrieved from https://planet.openstreetmap.org

[3] Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, "Exploring network structure, dynamics, and function using NetworkX", in Proceedings of the 7th Python in Science Conference (SciPy2008), Gel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11-15, Aug 2008

[4] Fruchterman, Thomas M. J.; Reingold, Edward M. (1991), "Graph Drawing by Force-Directed Placement", Software - Practice & Experience, Wiley, 21 (11): 1129-1164, doi:10.1002/spe.4380211102.

[5] Bastian M., Heymann S., Jacomy M. (2009). Gephi: an open source software for exploring and manipulating networks. International AAAI Conference on Weblogs and Social Media.