Copyright

by

Duncan C Anderson

2020

HIDDEN TEXT: Optional—If you do not include a copyright page, delete entire page and the following page break.

The Thesis Committee for Duncan C Anderson

Certifies that this is the approved version of the following <Thesis or Report>:

Application of Clustering Algorithms for Online Ridesharing and Infrastructure Placement in Transportation Networks

APPROVED BY

SUPERVISING COMMITTEE:

|  |
| --- |
| Dr. Steven Boyles, Supervisor |
| <Member’s Name> <, Co-Supervisor, if any> |

Application of Clustering Algorithms for Online Ridesharing and Infrastructure Placement in Transportation Networks

by

Duncan C Anderson

Hidden text: identical to name in UT transcript. Abbreviate previous degrees earned.

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Masters of Science in Engineering: Operations Research and Industrial Engineering

The University of Texas at Austin

December, 2020

Dedication

<Optional: If you do not include a Dedication, delete the entire page. **Do not delete the section break below.** It is needed to initiate the printing of page numbers from the Acknowledgements page on.>

Acknowledgements

<Optional: if Acknowledgements page is not used, delete the entire page, including the following page break. **Do not delete the section break above.** It is needed to initiate the printing of page numbers from this page on.>

Abstract

Application of Clustering Algorithms for Online Ridesharing and Infrastructure Placement in Transportation Networks

Duncan C Anderson, MSE ORIE

The University of Texas at Austin, 2020

Supervisor: Steven Boyles

<Abstract: Should not exceed 350 words. It should be a concise statement of the nature and content of the thesis or report.> Transportation network companies (TNCs), often called “ridesharing” companies, operating today face a number of challenges in implementing ridesharing in their networks. The most substantial of these is efficiently identifying high quality ridesharing opportunities in an online operating environment. Optimal ridesharing, also known as the multiple vehicle pickup and delivery problem, is NP-hard and must be solved approximately using heuristics in an operating environment where riders almost never place advance reservations. The quantity and diversity of origin and destination locations within an urban environment also create challenges for quickly evaluating ridesharing. Implementing pick-up and drop-off (PUDO) locations within the network can simplify ridesharing heuristics, allowing more accuracy in online ridesharing.

This research proposes the application of clustering algorithms on transportation networks as a solution to both of these problems. A ridesharing algorithm is proposed that uses a cluster adjacency index to quickly identify efficient ridesharing opportunities. The effectiveness of the ridesharing algorithm is examined using both K-means clustering, agnostic of network structure or congestion, and asynchronous fluid communities, a network clustering method that supports congestion weighting. The efficacy of the ridesharing algorithm is also assessed against different levels of clustering in the network.

Node clustering in the urban core is used to identify well connected areas of the network that can be served by a single PUDO. PUDO placement methods within the cluster that are both conscious and agnostic of congestion and network structure are examined.

The conclusion of this research is that properly calibrated network clustering methods with consideration of congestion are effective both for improving the quality of online ridesharing and for identifying PUDO locations.

## Table of Contents

HIDDENT TEXT: If you choose to place the chapter number (“Chapter 1”) and the chapter title (“Introduction”) on different lines, the automatically generated table of contents will reflect that format. After creating a new table of contents, set them on the same line by deleting the page number and paragraph marker at the end of each chapter number line.

Table of Contents viii

List of Tables ix

List of Figures x

List of Illustrations xi

CHAPTER 1: INTRODUCTION 12

CHAPTER 2: SIMULATION DESIGN AND PARAMETERIZATION 17

2.1 Network and Demand 17

2.2 Simulation Structure 21

2.3 Ridesharing 21

2.3 Congestion Simulation and Calibration 25

CHAPTER 3: CLUSTERING ALGORITHMS 27

3.1 K-means Clustering 27

3.2 Asynchronous Fluid Communities 28

CHAPTER 4: PUDO PLACEMENT 30

4.1 Centroid Based PUDO Placement 30

4.2 Centroid Based PUDO Placement 30

CHAPTER 5: SIMULATION CALIBRATION 32

5.1 Fleet Size Calibration 32

5.2 Cluster Size Calibration 35

CHAPTER 6: RESULTS 36

6.1 Impact of Well Connected Communities on Ridesharing 36

6.2 Well Connected Communities and PUDO Placement 36

6.3 Congestion Conscious PUDO Placement 36

CHAPTER 7: CONCLUSIONS 36

7.1 Conclusions of This research 36

6.2 Limitations and Further Research 36

## List of Tables

## No table of contents entries found.List of Figures

Figure 2-1 The Simulated Austin Texas Network 19

Figure 2-2 Network Abstraction in the Austin Texas Network 20

Figure 2-3 Austin Urban Core Nodes 21

Figure 2-4 Illustration of cluster based ridesharing methodology 24

Figure 2-5 Trip Demand by Minute for Full System and Urban Core 26

Figure 6.1 Dropped rides by hour in fleet size sensitivities 34

Figure 6.2 Total VMT by hour in fleet size sensitivities 35

## 

## CHAPTER 1: INTRODUCTION

A persistent or perpetual problem in the field of industrial engineering is the efficient movement of goods through a network. This broad class of problems traditionally considered the movement of physical freight or goods through distribution networks and manufacturing processes, but has expanded dramatically alongside modern computing and the discovery that many types of problems can be rephrased or reconstructed as network problems. In the modern era, much of this research concerns the movement of information through non-physical systems like social networks, and this research will borrow from techniques developed in that field. However, this research deals with one of the older applications of network theory, the challenge of moving people efficiently through a city.

Historically this field of research has been dominated by the public sector, municipalities concerned with the proper design of expensive public transit systems and the improvement and evolution of the road systems within the city to manage growing congestion. Taxis, the existing intra-city private transportation services, operated in tightly regulated markets for decades that left them mostly unconcerned with efficient or competitive operations (Salanova, et al. 2011). This balance has shifted dramatically with the advent of the transportation network company, or TNC (Wallsten, 2015).

TNCs like Uber and Lyft have created markets where private contracted drivers can offer door to door transportation services in their own vehicles, competing directly with existing taxi services and sidestepping many of the regulations and market structures that allowed taxi companies to operate uncompetitively. The effect of exposing these protected markets to competitive pressures has been paradigm shifting (Wallsten, 2015). In New York City the price of a taxi medallion, required to operate a taxi vehicle, has fallen more than 80% since 2013 (New York Times, 2017). Because TNCs only provide the service of connecting drivers with riders, and do not need to own vehicles or associated assets like garages, they face very few supply limitations to their growth, and have expanded rapidly over the last decade. Uber was founded in 2009 and just ten years later in 2019 serviced almost two billion trips. As these companies expand there is growing interest in methods that might increase the efficiency of their operations, driven both internally in response to competitive pressures and by municipal governments concerned about the congestion and road use impacts of ridesharing operations.

Some of the impacts of this TNC paradigm shift are good for municipalities and others are not. Commuters taking rideshares to and from work reduces demand for parking downtown, and a TNC driver can potentially meet the trip demand of multiple individuals that might otherwise drive personal vehicles, reducing the total number of vehicles on the road. There is some evidence that TNC costs per mile have dropped so low that they are cannibalizing demand for public transit systems, offsetting this effect (Graehler et al., 2019). TNC drivers waiting for or dropping off passengers can block and congest smaller roadways, and empty or unutilized TNCs often remain in or relocate to high density areas, further increasing congestion (Erhardt et al, 2019). TNC companies are also impacting the revenue that municipalities receive from on-street parking (Clark and Brown, 2020).

One proposed solution that cuts across all of these impacts is to designate pick up and drop off (PUDO) areas where these vehicles are allowed to service rides. PUDOs create an opportunity to utilize existing parking infrastructure, including lots, garages, and on-street parking. They eliminate pick-up and drop-off activities that obstruct traffic, and also provide a place for inactive vehicles to wait for riders without contributing to congestion on the roads (Chenqi, 2019). PUDOs have particular value, and already see some use by TNCs, around highly congested areas of major cities like sports arenas and weekend nightlife areas. They have also seen wide adoption in airports, where dramatic increases in ridesharing were creating congestion issues in existing drop off areas (Zuniga-Garcia and Machemehl, 2020). But beyond these limited applications the use of PUDOs has not been substantially explored by TNCs.

The concept of supply and demand aggregation points in door-to-door delivery is common in logistics from freight to telecommunications, where there is a well understood challenge called the “Last Mile Problem”. In package delivery its estimated that more than half the total shipping cost is accumulated in delivering goods the last mile to the home (Ranieri et al, 2018). For a TNC vehicle this last mile might involve exiting the highway, waiting at traffic lights, and navigating smaller congestible streets to reach an office space downtown. But unlike a package a human can walk this final distance, in some cases reaching the destination faster than a vehicle could. Depending on the pricing model used by the TNC, particularly the relationship between the flat fee, the cost per mile, and the cost per minute, it might be appealing to both parties to offload these “last mile” portions of the trip onto the rider.

The aggregation of riders at defined points may also increase TNCs ability to route trips efficiently and to provide ridesharing, where two unrelated trip requests with similar routes share a single vehicle for some portion of their route (Gurumurthy and Kockelman, 2020). Broad adoption of ridesharing would reduce externalities like congestion and emissions, but the nature of trip demand creates challenges for ride sharing adoption.

Trip demand is overwhelmingly on-demand and time sensitive. Riders rarely reserve trips in advance, trips must be served as soon as possible after they are requested and must be routed efficiently to minimize delays. This creates a very challenging environment within which to organize ridesharing opportunities. The core problem of ridesharing optimization, often referred to as the multi-vehicle pickup delivery problem (MVPDP), is NP hard. Without reservation or advance knowledge of trip requests ridesharing must be solved live while the system runs, and to avoid unacceptable delays in trip servicing it must be solved quickly using heuristic methods (Liu et al., 2018).

Optimal ridesharing is particularly challenging because of the size of the problem and the heterogeneity of trip requests. Routes very rarely begin and end at the same locations, and the number of nodes in the network, number of active vehicles, and volume of trip requests are all very large. Even metaheuristic methods for ridesharing that segment or abstract portions of the problem struggle to reach near optimal solutions in on-line timeframes (Liu et al., 2018, Alonso-Mora et al., 2017). Maciejewski and Bischoff (2016) are able to solve vehicle assignment in real time, but their system does not include ridesharing, which makes their solution a variant of the assignment problem rather than MVPDP. The practical result is that online ridesharing models are limited to sub-optimal solutions with significant detours that are often unattractive to riders.

One way to increase the speed of ridesharing algorithms is to reduce the size of the network by identifying groups of similar nodes and grouping them into a cluster. The quality of ridesharing algorithms that treat groups of nodes as a single node depends on how well connected the nodes are that form the cluster. This paper examines both simple and sophisticated methods for defining clusters within a network and attempts to quantify the impact of cluster quality and consideration of parameters like network structure and congestion. While exploration of on-line MVPDP is beyond the scope of this paper, a heuristic ridesharing method reliant on network clustering is operated in real time and demonstrated to capably handle hundreds of requests per minute in real time.

In addition to providing a high-quality abstraction of the network for ridesharing algorithms, clusters can also inform the placement of PUDOs in dense urban areas. This paper explores methods that use clusters to inform the placement of PUDOs that reduce travel time for vehicles without creating long walking distances for riders to reach their destination. The efficacy of each method is examined by feeding cluster indexes and PUDO assignments into a discrete event simulation designed to approximate the operations of a single large TNC company in the city. Critical variables examined include vehicle miles travelled, customer walking distance, and ridesharing as a proportion of total rides. This research aims to demonstrate that the application of clustering methods can allow for effective PUDO placement and high quality ridesharing heuristics that directly measure detour length to support real time on-demand ridesharing.

## CHAPTER 2: SIMULATION DESIGN AND PARAMETERIZATION

### 2.1 Network and Demand

This research focuses on the operations of a TNC across the Austin Texas 6 county region. The network used, seen in Figure 1, consists of 7,466 nodes and 18,703 directed edges and is sourced from the Transportation Network Modeling Committee (TRB-AEP40) github. The trip data is drawn from a 2006 Texas Department of Transportation GPS enhanced household travel survey and consists of 156,939 trips across a 24-hour period. These are the same trip tables used by the Capital Area Metropolitan Planning Organization (CAMPO) for their regional travel demand modelling. This is just a small subset of daily trips in Austin, which number in the millions, so this trip dataset is treated as the demand for this particular TNCs ridesharing services.



Figure 2-1 The Simulated Austin Texas Network

The origin and destination coordinates in this trip dataset are exact, while the six county Austin Texas network used in this simulation is a substantial abstraction of a much more complex road network. Trips must be fitted to existing nodes within the network, with riders assumed to walk the remaining distance to their precise destination. In many ways this abstraction of the simulation network causes the nodes to already function effectively as a PUDO. In areas outside the urban core, whole neighbourhoods and subdivisions are often represented by a single node in the network. Figure 2 below illustrates some examples of these neighbourhood nodes, in yellow, in South Austin.

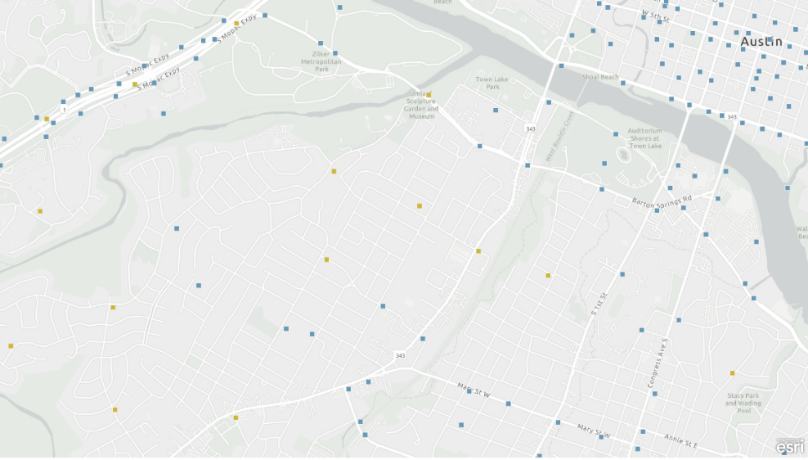


Figure 2-2 Network Abstraction in the Austin Texas Network

It’s not practical to cluster these nodes and choose a single PUDO, doing so would create inappropriately long walking distances for many riders. We define an appropriate walking distance as a quarter mile and find that a large number of nodes in the network, about 3200, are not suitable for PUDO assignment of any kind based on this threshold. Of the remaining nodes, many are node clusters associated with highway exits, frontage roads, and turnaround areas which contain tightly clustered road intersections. A comparison of trip origins and destinations with PUDOs generated across the system indicates, intuitively, that these are very rarely trip origins or destinations.

If the bulk of the nodes in the simulation network already function effectively as PUDOs, and many of the remaining nodes are not appropriate locations, then an examination of PUDO placement methods must be concentrated in the remaining area which has both high node density and contains a large number of trip origins and destinations. This area is often referred to as the ‘urban core’ or the downtown area of the city (Fagnant and Kockelman, 2014). This area, consisting of 229 nodes and 679 edges, can be seen in the figure below. While this represents only 3% of the network, more than 13% of trips begin or end in this area. The results section focuses on impacts to these trips from the application of various PUDO placement methodologies to this reduced population of nodes, however all trips across the full network are simulated to ensure realistic behavior for the fleet of TNC vehicles. Because clustering must be done across the full network, the exact population of nodes in the urban core region shifts slightly depending on which clustering method is used.

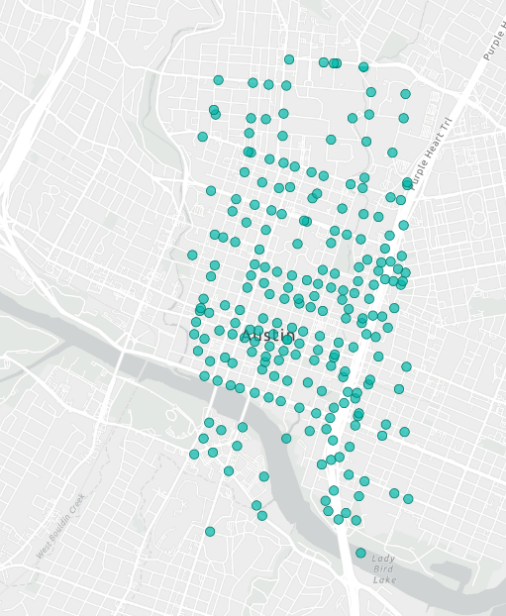


Figure 2-3 Austin Urban Core Nodes

### 2.2 Simulation Structure

The simulation is a discrete event simulation (DES), an efficient method which considers only events which impact the state of the system without consideration of the intervening time periods. The DES is built atop the NetworkX graph library for python, and routing within the network is handled using the A\* algorithm. There are four types of events that the simulation handles: ride requests, arrivals, rideshare arrivals, congestion events, and relocations. The bulk of the simulation is concerned with the handling of ride request events, which trigger ridesharing logic, handle routing and rerouting, and generate arrival and ridesharing arrival events. Ride request events reduce vehicle capacity at the origin location and arrival events increase vehicle capacity at the destination, apart from ridesharing arrivals which are intermediate drop-offs where the vehicle continues on its route. Congestion events are two minute events that temporarily increase the travel time on a link when a vehicle is picking up or dropping off riders.

### 2.3 Ridesharing

Because of the computational challenge of MVPDP, there are a broad variety of ridesharing heuristic methods developed in the literature, many more than can be detailed in this research. Many of these methods are offline, meaning their computation requires more time than is practical in an on-line operating environment, and so are not considered for comparison here. Gurumurthy and Kockelman describe a unique methodology in their 2020 paper that involves comparing a calculated Euclidean angle of directionality for a route and a proposed detour to a defined threshold to ensure routes are traveling in similar directions (2020). Schreieck et al. implement an inverted index structure for storing the node components of ongoing rides to allow them to efficiently search them each time a proposed rideshare becomes available, though the method is not simulated (2016). Alonso-Mora et al. (2017) implement a greedy ILP that can be solved in real time by terminating the ILP at intermediate sub-optimal solutions. While its possible this ILP might approach optimal MVPDP solutions given enough solve time, solution in real time guarantees sub-optimal solutions, and its disputable whether there is a practical linear relationship between runtime and solution quality for an ILP of this structure, the solutions are only guaranteed by the authors to be better than a greedy heuristic used to initialize the algorithm.

The ridesharing logic in this simulation is also heuristic and is made computationally tractable in real time by the clustering methods described in the next section. Runtime considerations often prevent heuristic ridesharing algorithms from using on-network routing to evaluate potential rideshares, and reliance on precomputed distance tables, layered route indices, or other data structures is common to avoid large volumes of shortest path calculations to evaluate detour alternatives (Schreieck et al., 2016). By clustering the nodes adjacent to a route into “well connected communities” and indexing their adjacencies the simulation used here can easily identify ridesharing opportunities that involve only efficient detours from its current route. Clustering methods that fail to consider network structure are also explored and still allow for rapid identification of nearby ridesharing opportunities, but sometimes create inefficient detours through congested areas.

In order to assess ridesharing opportunities for a ride a list of all clusters the ride passes through and their adjacent clusters is constructed. Then a list of rides currently underway (enroutes) is processed using the current simulation time to determine the position of each enroute vehicle and the remaining clusters that it will pass through or adjacent to. A reduced list is constructed of only enroutes where the origin cluster of the ride request is within or adjacent to the routes remaining cluster trajectory. From this list a further reduced list of enroutes is created where the enroute destination cluster is within or adjacent the trip cluster trajectory, or the trips destination cluster is within or adjacent to the routes cluster trajectory. If this final list contains more than one route the detours are calculated for each route and minimized. Figure 4 contains several visualizations that illustrate how this logic functions.

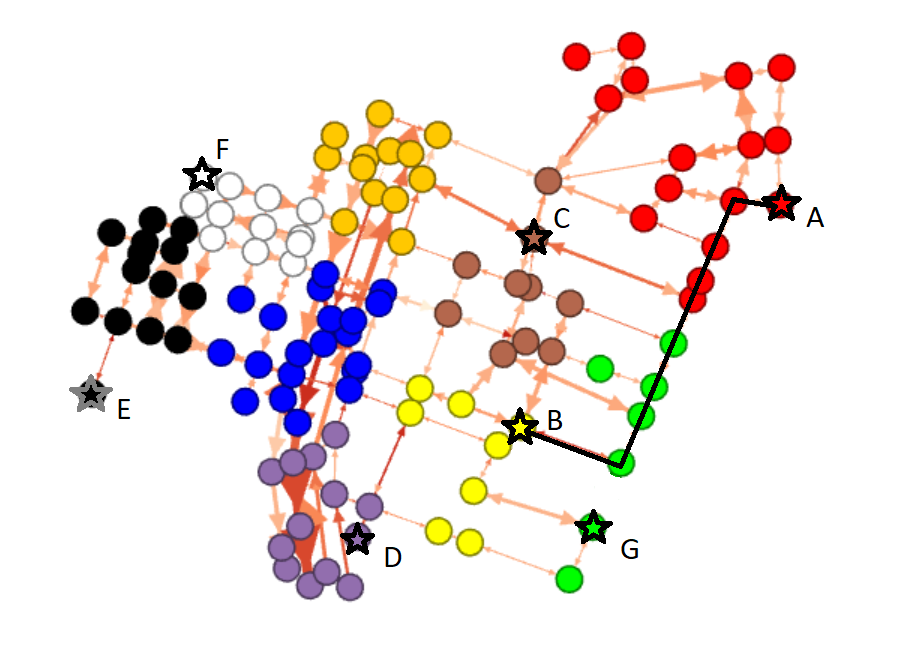


Figure 2-4 Illustration of cluster based ridesharing methodology

In Figure 4 above, there is an ongoing ride currently node A travelling to node B, its route is marked by the black line. A rideshare request originating at node E or F would not be considered, because the clusters these nodes are in, black and white, are not adjacent to the current route. A ride originating at node D would also not be considered, because the purple cluster is only adjacent to the final cluster of the ongoing ride, and the logic avoids detours near the ride destination. A ride originating at C in the brown cluster would be considered initially because its cluster is adjacent to the clusters of the ongoing route, but eligibility is also dependent on the rides destination. C to G would be an accepted enroute rideshare, because G is in the yellow cluster which is a component of the current routes clusters. As an enroute rideshare the rider at C would be dropped at G before the current rider is dropped at B. C to D would be an accepted rideshare if the route from C to D passes through the current destination yellow cluster, and would be considered an extension rideshare, the current rider would be dropped at B before the vehicle continued to its final destination at D. C to E would similarly only be eligible if the A\* route passes through the yellow cluster. C to F would likely not be an eligible rideshare. By inspection any ride originating at G would be eligible for ridesharing, because the A\* route to any lettered node must pass through yellow, the current destination cluster, and the green origin cluster is on the current routes cluster list.

In this way, the clustering method has a direct impact on ridesharing. Because the quality of the clustering method dictates how well connected the communities are that the vehicle travels through, a higher quality cluster reduces the chances of the ridesharing logic producing an inefficient detour. In the example above node G, has been clustered with nodes on the current route but has no interconnection to these nodes. So rides originating at G are eligible rideshares for the current route, even though the detour to reach G is substantial.

The ridesharing methodology is also sensitive to the number of clusters defined in the network, as lower numbers of clusters increase the number of nodes that share a cluster or adjacency with a given enroute. So it is important to control the number of clusters created across comparative simulations, and also to evaluate the level and quality of ridesharing at different cluster sizes. A more thorough examination of this problem is included in the model calibration section.

### 2.3 Congestion Simulation and Calibration

To assess congestion on the network the trip dataset is scaled up and static traffic assignment is performed on the network using provided link performance function parameters for each edge. Congestion is assessed for three distinct periods during the day: The morning traffic peak from 7:00 AM to 9:00 AM, early evening rush hour from 4:00 PM to 7:00 PM, and all other hours.

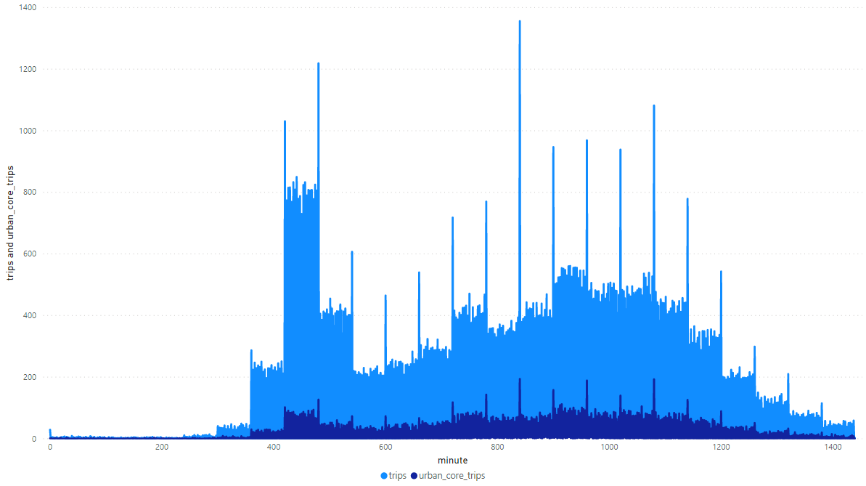


Figure 2-5 Trip Demand by Minute for Full System and Urban Core

Static traffic assignment is done using the Frank Wolfe algorithm for each of these periods. Real travel time data for each edge during the morning peak was collected from the Google Maps API. Total trips are iteratively scaled up, and the resulting travel times compared against real travel time data. For each iteration a usage weighted root mean square error (RMSE) was calculated for each edge, representing the error between actual and simulated travel time weighted by the number of trips using that edge. The trip scaling level with the minimum weighted RMSE was selected.

It was observed that the error rates for this methodology were very high, and that the congestion levels resulting from static traffic assignments on the network were not similar to the actual congestion levels gathered from the google maps API. There are a number of factors that could contribute to this, including coordinate imprecision, changes to the real traffic system since the simulation graph was developed, and uncertainty about the quality of the sampled trip data or limitations in scaling it to reproduce full traffic conditions.

As mentioned in the simulation section vehicles picking up or dropping off riders generate a congestion event in the schedule, impacting the congestion on their current edge for three minutes. The magnitude of this event is dependent on the alpha, beta, and capacity values for the edge that compose the BPR link performance function, the effect is equivalent to the addition of a number of vehicles equal to approximately 5% of the capacity of the road.

## CHAPTER 3: CLUSTERING ALGORITHMS

### 3.1 K-means Clustering

This research explores two distinct methods for clustering nodes in the network to evaluate the importance of well-connected community identification. The first method used to identify clusters is normalized k-means. K-means groups nodes into clusters called Voroni cells and then measures the Euclidean distance from the calculated centroid of each cluster to all the nodes in that cluster. The algorithm iteratively shifts nodes between clusters until it has minimized within-cluster variance to a specified threshold. This research examines both unweighted k-means clustering of network nodes by normalized coordinates, and trip weighted clustering where nodes are weighted by how often they appear as an origin or destination in the trip data. In both cases the PUDO for a cluster is placed at the node closest to the calculated centroid, the location closest to all nodes in the cluster by Euclidean distance.

K-means has several drawbacks that limit its application. The first is that there is no consideration of network structure in forming the clusters. This can lead to inclusion of nodes in a cluster that are not well connected to the other nodes in the cluster, creating long walk distances for rides that begin or end at badly connected nodes, and similarly inefficient ridesharing detours. The second drawback is that absent consideration of network structure, the algorithm has no ability to consider congestion in its assessment of clusters. Addressing these concerns requires a cluster or community detection method that considers the characteristics of the network.

### 3.2 Asynchronous Fluid Communities

Design of effective clustering methods for directed graphs is a subject of debate in the literature. The majority of literature around graph clustering focuses on undirected graphs, and much of the literature detailing methods for clustering directed graphs acknowledges that its common to treat edges as undirected for the purposes of community detection (Santos, 2016). In particular, the idea of clusters or well-connected communities becomes complicated when directional edges must be considered (Malliaros and Vazirgiannis, 2013). Many of the proposed methods for clustering congested networks don’t allow specification of the number of clusters, which creates challenges the comparative analysis in this research (Fagiolo, 2007). Fortunately the cluster definition needs for this research are mostly directionally agnostic. Vehicles and riders must flow both in and out of PUDOs, and enroute vehicles must detour out and back across clusters to pick up rideshares, though often not on the same edges. For this reason, cluster definition is done using an undirected graph with the edges weighted in both directions with their greatest (more congested) value.

The algorithm applied to this undirected congestion graph is asynchronous fluid community detection, developed in 2017 at the Barcelona Supercomputing Center (Pares et al., 2017). Asynchronous fluid communities is a label propagation method based on the movement of fluids, characterized by their component nodes, pushing on each other as a result of their characteristics and the topology of the network environment. One critical characteristic of this detection method is that it is the only label propagation method that allows for a specific number of communities to be generated. Like a fluid spreading over a surface, the fixed volume limits how far it can spread. This fluid volume can be precalculated to allow any number of communities to be defined within a network. This allows us to directly compare asynchronous fluid communities to k-means clustering and isolate the impact of well-connected communities, without worrying about the impacts of mismatched cluster sizes.

## CHAPTER 4: PUDO PLACEMENT

### 4.1 Centroid Based PUDO Placement

In all scenarios in this paper clustering methods are applied across the entire network in order to facilitate efficient ridesharing and vehicle reallocation, but within the urban core of the network the identified clusters or communities are used as the basis for PUDO creation and placement. Two distinct PUDO placement methods are explored. The simpler method involves the calculation of a centroid for each cluster, a point which minimizes distance between it and all nodes in the cluster. This centroid is then matched to its nearest node in the network and the PUDO is placed there. The benefit of this approach is that it minimizes the walking distances required to the extent this is possible without advance knowledge of trip demand. The limitation of this approach is that these well connected routes within the community are rarely edges where replacing driving with walking creates efficiency.

### 4.2 Centroid Based PUDO Placement

The second method of PUDO placement involves a more nuanced understanding of the role of PUDOs in the system, and access to good information about congestion levels on the edges in the network. PUDOs are ideally placed adjacent to congested edges, where riders walking a final edge or two to their destination is preferable or comparable to driving. Simultaneously it is critical that the PUDO have both inflow and outflowing adjacencies that are not congested, in order to handle the heavier vehicle traffic from the aggregation of rides at the location. An algorithm to assess a nodes fitness for a PUDO is proposed as:

1. For each node in a cluster, create list of 2nd degree adjacent nodes – all nodes adjacent to nodes adjacent to the primary node.
2. Create a list of 2 node routes that lead in and out of the node.
3. For each route, define a congestion ratio as the morning peak congested travel time divided by the uncongested travel time
4. For each node, assess the difference between the maximum and minimum congestion ratios
   1. A high congestion ratio marks a route to the node that experiences significant congestion
   2. A low congestion ratio marks a route to the node that does not congest
5. In each cluster, the node with the highest spread in its congestion ratios is chosen for PUDO placement.

Well-connected communities in a network will rarely contain heavily congested edges, so this method tends to place PUDOs at the edges of well-connected communities, and creates many situations where the nearest PUDO to a trip origin is technically in an adjacent community.

## CHAPTER 5: SIMULATION CALIBRATION

### 5.1 Fleet Size Calibration

The number of operational vehicles in the simulation can meaningfully impact simulation outcomes. Too few vehicles and a substantial portion of rides, particularly rides during peak periods, are dropped. Initializing too many vehicles in the simulation detracts from ridesharing and diminishes the importance and impact of fleet positioning on results. In order to assess the appropriate number of vehicles a series of fleet size sensitivities are run from 7,500 to 15,000 vehicles on a network clustered into 2,000 asynchronous fluid communities. While the fleet size requirements are likely dependent on the quality and quantity of clusters used, this cross-parameter sensitivity analysis outside of the scope of this paper. The results of the fleet size sensitivities are shown in the figure below.

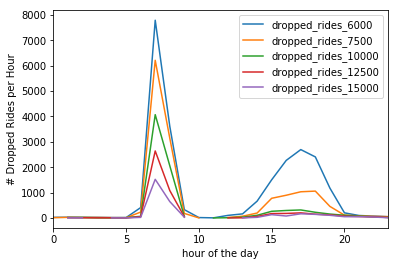


Figure 6.1 Dropped rides by hour in fleet size sensitivities

At a fleet sizes of 12,500 vehicles the TNC drops rides almost exclusively during the morning peak period, and drops about 3% of total rides. At smaller fleet sizes dropped trips begin to appear in the evening peak and the number of dropped trips increases dramatically to about 10% of rides with 7500 vehicles and about 15% of total rides with 6000 vehicles.

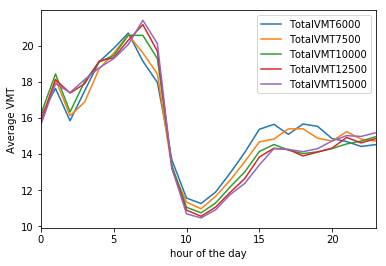


Figure 6.2 Total VMT by hour in fleet size sensitivities

Examining variables like VMT and average ride time is complicated by long rides being dropped during peak period, so its critical to evaluate these variables outside of peak periods. If we look at the hours after the morning peak we can see that both the 6000 and 7500 vehicle sensitivities significantly increase average VMT and have even larger proportional impacts on ride time. Both dropped rides and average VMT are critical quality indicators for TNCs, and while some amount of each is inevitable, particularly during peak periods, the fleet should be sized appropriately to prevent significant impacts to either during below-peak hours.

Fagnant and Kockelman (2015) use a fleet of 2000 vehicles to serve 56,000 trip requests drawn from this same trip data over 24 hours, which implies an appropriate fleet size of about 6,000 vehicles for this TNC. But Fagnant and Kockelman examine only trips with both origins and destinations within a twelve square mile radius of the urban core. This substantially reduces travel time during peak periods by eliminating long commute trips, allowing vehicles to complete multiple trips during the morning peak. Boesch et al, 2016 suggests a threshold of 95% of trips served within 5 minutes, which aligns with our 10,000 vehicle sensitivity. Bischoff and Maciejewski simulate different fleet sizes dispatched against travel demand in Berlin and accept a large percentage of dropped rides, about 20,000 or 17% of travel demand, at its peak in the afternoon (Bischoff and Maciejewski, 2016). Loeb and Kockelman (2019) drop 16.2% of trips in their short range SAEV scenario, and 15.2% of trips in their long range SAEV scenario. Both are both roughly aligned with our 10,000 vehicle simulation, which drops about 19% of rides during the morning peak hour. Our 7,500 vehicle simulation drops closer to 30% of rides during this period, and about 14% overall, so we select 10,000 vehicles as the appropriate fleet size for our simulations.

### 5.2 Cluster Size Calibration

The methodology for assessing potential rideshares in this model relies on cluster definitions rather than defined detour thresholds, so a similar calibration sensitivity exercise must be performed for cluster size in order to properly calibrate cluster dependent ridesharing. Too few communities will produce large clusters along and adjacent to a route and could create long or inefficient detours that increase VMT. Too many communities will restrict the ridesharing algorithm to narrow consideration of potential rideshares directly along its route, which decreases ridesharing and increases total VMT. These calibration runs use asynchronous fluid community generation and cluster counts from 1000 to 3000. For each sensitivity we look at the impact on ridesharing detours. For both the ongoing ride and the rideshare request, we look at the difference between its realized route in the simulation and an initial direct route computed when the ride is first hailed. The quality of the ridesharing in each of the scenarios is examined through average detour duration and the average detour length. We also look at total VMT across the scenarios.

## CHAPTER 6: RESULTS

### 6.1 Impact of Well Connected Communities on Ridesharing

### 6.2 Well Connected Communities and PUDO Placement

### 6.3 Congestion Conscious PUDO Placement

## CHAPTER 7: CONCLUSIONS

### 7.1 Conclusions of This research

### 6.2 Limitations and Further Research