Vehicle Relocation for Ride-Hailing

Joon-Seok Kim
Geography and Geoinformation Science
George Mason University
Fairfax, VA, USA
jkim258@gmu.edu

Dieter Pfoser
Geography and Geoinformation Science
George Mason University
Fairfax, VA, USA
dpfoser@gmu.edu

Andreas Züfle
Geography and Geoinformation Science
George Mason University
Fairfax, VA, USA
azufle@gmu.edu

***Abstract*—Ever increasing traffic and consequential congestion wastes fuel and is a significant contributor to Green House Gas (GHG) emissions. Contributors here include ride-sharing services such as Uber, Lyft, and Didi, with their drivers not only transporting passengers, but also spending a considerable time in traffic searching for new ones. To mitigate their impact, this work proposes a novel algorithm to improve the efficiency the drivers' search for passengers. Our algorithm directs unassigned drivers to locations where new passengers are expected to emerge. We use a non-negative matrix factorization approach to model the time and location of passengers given historical training data. A probabilistic search strategy then guides drivers to nearby locations for which we predict new passengers. To ensure that drivers do not over subscribe to such areas, we randomize destinations and provide each driver with a home location destination when unassigned. An experimental evaluation using real-world data from Manhattan shows that our approach actually reduces the search time of drivers and the wait time of passengers compared to baseline solutions.**

***Index Terms*—Non-negative matrix factorization, simulation, discrete event simulation, spatiotemporal search**

I. Introduction

In the United States alone, drivers spend 6.9 billion person-hours stuck in traffic each year [19], leading to an annual waste of more than 11 billion liters of fuel [10]. The transportation sector alone contributes 29% of the total US Green House Gas emissions, of which in turn 59% are contributed by passenger vehicles [17]. Adding to this balance is the increasing popularity of ride-hailing services such as Uber, Lyft, and Didi. Recent research has shown that ride-hailing causes a vast increase in vehicle kilometers traveled [21], thus worsening traffic conditions. This result is intuitive, as using such services incur travel not only during a trip but also to pick up passengers and drivers *cruising* in search for the next passenger. In the future, a fleet of autonomously driving ride-sharing vehicles will worsen this situation as predicted by Bryan Mistele, founder and CEO of INRIX in a SIGSPATIAL 2017 keynote[1].

The aim of this work is to help mitigate this emerging problem by providing smart strategies and heuristics for drivers to intelligently roam the streets while searching for passengers to minimize the average time of drivers without passengers (denoted as *search time*). We assume that drivers do not know the location of waiting passengers unless the passenger is assigned to them. This scenario reflects current ride-sharing applications such as Uber, Lyft, and Didi, which give passengers information about the location of nearby drivers, but do not show drivers the location of nearby passengers. The milestone reached in this work is to help drivers make a more educated guess and to reduce resources (time, fuel, traffic capacity) spent by drivers cruising in search for passengers. Unlike previous work, we assume that we can control the direction of drivers when not assigned to a passenger. Therefore, our goal is not to match drivers to passengers, but rather, to smartly move idle drivers through the network to achieve a better assignment in the future. Towards this goal, we leverage an agent-based simulation framework, combined with available real-world passenger data (origin and destination) and employing a non-negative matrix factorization approach to model and predict the distribution of passengers over space and time. Given this prediction of passengers and their locations for a limited time horizon, we provide a strategy to guide idling drivers to optimize future assignments.

A four-page invited (non-peer reviewed) workshop version of the idea for this algorithm has been previously published [15]. Towards making the leap from an algorithm to a system, the goal of this paper is to include justification for technical approaches and parameter settings, provide technical details for reproducibility, and discuss challenges towards the deployment of our algorithm in a ride-hailing system. We also include a thorough experimental evaluation, showing its superiority to baseline strategies.

These contributions are organized as follows. We first embed our approach in the context of existing work in Section II. Section IV motivates the rationale by visually exploring a simulation of drivers in New York City using real passenger data. Section V-A describes the non-negative matrix factorization approach to model and predict passenger demand over space and time. Using this model, the search algorithm to guide drivers is detailed in Section V-B. The experimental evaluation of Section VI shows that our algorithm reduces the average search time of drivers and the average wait time of passengers compared to baseline solutions. Towards adoption of our algorithm by a ride-hailing service provider, we discuss limitations due to simplifying assumptions made in Section VII. Finally, we conclude our work in Section VIII.

[1]https://sigspatial2017.sigspatial.org/keynotes/#bryan

II. Related Work

Our survey of related work addresses task assignment in spatial crowdsourcing and spatiotemporal resource search. We also discuss existing work on simulations for ride-hailing applications and review existing solutions for competitive spatiotemporal searching.

A. Spatial Crowdsourcing

Related to competitive spatiotemporal searching is the problem of task assignment in spatial crowdsourcing. In spatial crowdsourcing, each crowd worker (driver) is considered as a mobile computing unit to complete tasks (reach resources) using their mobile devices [22], [16]. Task assignment aims to assign tasks to workers such that the total number of assigned tasks or the total weighted value of the assigned pairs of tasks and workers is maximized. Highly efficient solutions for this problem have been proposed to derive a bi-partite matching between workers and tasks for the case in which all tasks (their time and location) are known in advance [11], [14]. Clearly, for the task ride-hailing, neither availability of drivers nor resources (passengers) can be assumed to be known in advance. For the online-case of spatial crowdsourcing, where drivers and resources are unknown beforehand, assignment solutions have been proposed [20], [23]. However, spatial crowdsourcing assumes that the location of workers cannot be changed. Thus, spatial-crowdsourcing is exclusively an assignment problem, without any notion of moving workers to locations where future resources may appear. In contrast, this work considers the generalized problem of re-positioning drivers during their idle time towards areas having a high chance of yielding future resources.

B. Spatiotemporal Resource Search

Ayala et al. [3], [4] introduce parking slot assignment games to analyze parking slot search in a competitive setting. Similar to spatial crowdsourcing solutions, the goal is to find an optimal assignment of cars to parking lots, since each car is removed from the system once a parking lot is found, without the need to find more resources. A more general version of this problem is the spatiotemporal resource search problem [12], in which mobile agents (drivers) minimize search time for finding resources (passengers) similar to our setting. In our problem, however, we do not aim to optimize for an individual driver, and rather optimize a system in which drivers continuously seek resources rather than searching for a single resource. Recent work in operations research [5], [6] has considered the problem of dynamic scheduling for independent, competing taxis. This work focuses on assigning taxis to passengers in an optimal way. This work is a special case of the vehicle routing problem [18]. Most related to our work is a system, deployed by DiDi [24], which dynamically assigns drivers to new passengers. All aforementioned works have in common that the goal is to optimally assign drivers to passengers. In our setting, we assume that an assignment strategy is given by simply pairing taxis and passengers based on nearest neighbors. Our goal is not to match taxis and passengers, but to distribute unassigned taxis on the network, in such a way that future passengers can be assigned more efficiently. None of the aforementioned works consider strategies for drivers to change their location between trips, as each mobile agent (driver) is assumed to be terminated once a resource (passenger) has been found.

C. Traffic Simulation for Ride Hailing

Geospatial simulation approaches have been proposed to evaluate how different strategies to assign drivers to passengers have different effects on passenger wait time, reduce system wide travel times, and maximize revenue of the ride hailing service provided [2]. A more in-depth study was performed to see how ride-hailing affects traffic conditions, by estimating the increase of distance traveled replacing the use of personal cars with ride-hailing [21]. While our approach also uses geospatial simulation, we again note that our approach does not aim at optimizing driver-passenger assignment but, rather, aims at providing heuristics to drivers that are currently unassigned to improve future assignments.

D. Competitive Spatiotemporal Searching

The problem of vehicle relocation for ride-sharing, which is addressed in this work, has recently been defined as a GIS-focused algorithm competition part of the ACM SIGSPATIAL 2019 GIS Cup[2]. Similar to Spatiotemporal Resource Searching, the goal is to assign passengers to drivers such that the average idle time of drivers (without being assigned to a resource) is minimized. In Spatiotemporal Resource Searching the goal is to find an optimal assignment strategy for a given search strategy for drivers. In contrast, the problem of vehicle relocation is to provide a movement strategy for idle drivers, given a pre-defined assignment strategy that assigns each resource (passenger) to their nearest driver subject to a maximum range. Due to the uncertainty of resource distributions and drivers' behaviors, it is a challenge to manage such uncertainty in evolving spatiotemporal data [25].

Teams having the best results in terms of minimizing search times were invited to publish their solution in four-page non-peer reviewed short papers [13], [7], [15], [8]. The work of [13] employs a k-means clustering approach to partition space into areas onto which drivers are distributed, weighted by the observed passenger popularity. The approach of [7] uses reinforcement learning to compute round trips for drivers to optimally cover areas weighted by passenger distribution. A weighted random walk is proposed by [8], which sends drivers to random destinations weighted by the learned passenger distribution. Finally, the winner of GIS Cup 2019 [15], upon which our work is based on, uses a similar approach, but constraints the set of possible paths by a distance threshold to avoid long paths, and uses matrix-factorization to obtain a more accurate passenger distribution. In this work, we extend the work of [15] by providing more advanced search strategies, a more thorough experimental evaluation, and details for reproducibility.

[2]https://sigspatial2019.sigspatial.org/giscup2019/

III. Problem Definition

We want to formally define the problem of competitive spatiotemporal searching and review the open-source simulation framework *COMSET* that we leverage to study different behaviors of drivers while not assigned to a passenger.

Definition 1 (Road Network): A road network is a graph $\mathcal{G} = (V, E)$, where vertices in V are the intersection of roads, and edges in $E \subseteq V \times V$ are undirected road segments connecting intersections. Each vertex $v \in V$ has a geolocation $v.l$ and each edge $e \in E$ is assigned a travel speed $e.speed$. A location in a road network is a pair $(e \in E, offset \in [0, 1])$ that specifies the relative position on an edge between two vertices.

We assume a dataset of resources corresponding to passenger trips. Each resource has an origin (pick-up) location, a destination (drop-off) location, and a pick-up time upon which the resource appears (the ride-hailing service is requested).

Definition 2 (Resource Dataset): Let $\mathcal{T} = \{T_1, ..., T_{|\mathcal{T}|}\}$ be a domain of temporal intervals (or "ticks"). A resource r is a tuple $(l_o, t_o \in \mathcal{T}, l_d \in E)$ consisting of an origin location l_o and a destination location l_d. Time t_o defines the time at which the resource appears. A resource dataset $\mathcal{D}$ is a collection of resources.

We use the term "resource" instead of "passenger" to be consistent with related work on spatial crowdsourcing. While our work is mainly motivated by ride-hailing, our definition also applies to applications such as food delivery and other logistic applications that may not involve passengers. In addition, we assume a set of mobile agents (drivers) that traverse the road network searching for resources.

Definition 3 (Mobile Agent): A mobile agent a traverses the road network at a speed defined by the travel speed $e.speed$ of the current edge of a. A mobile agent has two possible states:

1) An agent can be unassigned. In this state, the agent follows a strategy $\mathcal{S}$ to traverse the network.
2) An agent may be assigned to a resource $(l_o, t_o \in \mathcal{T}, l_d \in E) \in \mathcal{D}$. In this state, the agent will take the shortest path to location l_o to pick the passenger, and then take the shortest path from l_o to l_d to drop off the passenger. Upon reaching l_d, the agent returns to the unassigned state.

Initially, at time T_1, all mobile agents are unassigned.

The ride-hailing provider assigns agents to resources. We assume a simple strategy that assigns a passenger resource to the nearest agent subject to a maximum distance.

Definition 4 (Agent-Resource Assignment): For assigning resources to agents, a global variable δ_{exp} is defined to denote that expiration time or the lifetime of a resource. Any resource that is not assigned within δ_{exp} is removed from the system. Any unassigned resource r is assigned to the closest available (unassigned) agent a if and only if the time required for a to reach r is less than the remaining expiration time of r. If no such agent exists, then the resource is not assigned. Note that an unassigned resource may still be assigned later, as a sufficiently assigned agent may drop-off a resource nearby, thus becoming available. When a resource r is assigned to an agent a, it is removed from the system, and a will transition to the *assigned* state. Any resource that is not assigned after δ_{exp} is removed from the system without assignment.

Given all previous definitions, we can proceed to define the problem of competitive spatiotemporal searching formally.

Definition 5 (Competitive Spatiotemporal Searching): Given a road network $\mathcal{G}$, a resource dataset $\mathcal{D}$, a set of mobile agents A and a resource assignment strategy having expiration parameter δ_{exp}, the problem of competitive spatiotemporal searching is to find a search strategy $\mathcal{S}$ for agents that minimizes the average agent search time

$$\sum_{t \in \mathcal{T}, a \in A} I_{\mathcal{G},\mathcal{D},\delta_{exp}}(\text{unassigned}(a, t))/|A|,$$

where $I_{\mathcal{G},\mathcal{D},\delta_{exp}}(\text{unassigned}(a, t))$ is an indicator function that returns 1 if agent a is unassigned at time t. To create a setting comparable to real-world ride-hailing applications, we further assume that search strategy $\mathcal{S}$ must not access (does not know) the current location of resources, and agents are not allowed to communicate/collaborate.

As an example, a naive search strategy $\mathcal{S}_{\text{random}}$ lets unassigned agents implement a random walk on the network, thus choosing a random (chosen uniformly) edge to follow at each vertex of the road network until the agent is assigned to a resource. Another naive search strategy will send agents on a shortest path to random destinations until they are assigned to a resource. Our strategy to guide agents is described in Section V. To evaluate different search strategies $\mathcal{S}$, we leverage a discrete-event simulation framework described as follows.

Simulation Framework

To evaluate different search strategies, we make use of the open-source simulation framework called *COMpetitive SEarching Testbed (COMSET)*[3]. COMSET uses a road network (as in Definition 1) from Manhattan, New York City, USA. Resources are streamed into the simulation, using origin location, destination location and time of appearance (see Definition 2)) taken from the New York City Trip Record Data [1] dataset. All mobile agents (as in Definition 3) are introduced at the beginning of a simulation, initially located at a random location on the road network. To assign resources to agent, COMSET uses the assignment strategy described in Definition 4 having a resource expiration time of $\delta_{exp} = 10min$.

IV. Visual Data Analysis

To motivate the rationale behind our competitive spatiotemporal search algorithm introduced in Section V, this section presents qualitative results that showcase the availability of drivers and resources over time using real-world data. We

[3]https://github.com/Chessnl/COMSET-GISCUP

(a) 9:10 am: Undersupply (b) 10:30 am: Bad Agent Distribution (c) 12:00 pm: Oversupply

Fig. 1: Several screenshots from the simulator (pink dot: available resource, blue dot: searching agent)

added a visualization layer to COMSET to show a road network as well as available resources and searching agents. Figure 1 shows searching agents as blue dots, while available resources are shown as pink dots. This figure shows snapshots of the simulation using a simple heuristic to guide drivers to a random new destination across the network until they are assigned to a resource. A video that demonstrates this visualization continuously for a simulation day is found at *https://sites.google.com/view/dsaa-2020* and is summarized in the following. For comparison, this video (but not Figure 1) concurrently shows the simulation using our proposed competitive spatiotemporal resource search algorithm proposed in Section V.

Undersupply: During the rush hour peak we observe an *undersupply* as shown in Figure 1a, which shows the simulation at 9:10 am on January 21st, 2016. At this time, resources become available at such high frequency that drivers cannot possibly keep up. During this time, drivers that finish a ride are immediately assigned to their next resource. We observe many available resources near central park, while only a few resources are available on the northern and southern side of Manhattan. Only very few drivers find themselves searching in the latter resource-deserts with no available resource in range. But, keep in mind that some of these agents have not been taken to this desert by their choice, but rather, were taken there by a destination of another trip. These cases yield a situation of "anywhere else is better than here," where even a simple baseline that guides drivers to a random destination is likely to lead a driver out of the desert. We note that in this case of having an undersupply of drivers, there is little potential to improve the search times of drivers, as most drivers are instantaneously assigned to new passengers, yielding an average search time that approaches zero.

Balanced supply: At 10:30 am on the same day, we observe in Figure 1b that resources are still available near Central Park. This implies that there is no available driver to assign to the resources. At the same time, we also see that agents are searching in other parts of the network. This is an interesting because, in this case, it is possible to achieve a potential improvement over this naive random-destination baseline! If some of the searching agents were close to the available resources, they could be assigned and would not have to search any longer. Thus, to make an improvement, searching agents need to be led to areas where resources are expected to become available, rather sending them to random destinations. We note that our video (*https://sites.google.com/view/dsaa-2020*) shows qualitatively that during this time of 10:30 am our approach (denoted as 'Local Search' in the video) yields a much better assignment from resources to drivers: Much fewer resources are waiting for drivers, and much fewer drivers are searching for resources. This is achieved by ensuring that drivers are likely to be located where they will be needed as described in Section V and quantitatively evaluated in Section VI.

Oversupply: The third case that can be observed in this simulation is *oversupply*, as shown in Figure 1c during noon on the same day. During this time, resources become so scarce that it is impossible to assign each agent to a resource. We cannot see any available resources, as any new resource is immediately assigned to a driver in range. However, we see many drivers searching. While our goal is to minimize the search time for agents, we must acknowledge that it is not possible to avoid search times during oversupply. However, what we can do is to make sure once the demand of passengers increases, our drivers are readily waiting in the right locations by predicting future demand but while also avoiding "herding" too many drivers into the same area. This observation is the main motivation of our algorithm described in the next section.

To summarize the lessons learned from data visualization, our only hope of reducing the search time is to reduce search times is during times when there are searching agents in some parts of the network, while there are resources available in

other parts of the network. Towards this goal, we need to predict these areas of high resource availability, and we have to direct searching agents to find these resources quickly. In our video, you can further see that our proposed algorithm is seemingly perfectly able to supply all resources with agents. There are few times in which both waiting resources and searching drivers can be observed at the same time. In the video, you can see that during the afternoon rush at around 5:50 pm, the network is nearly empty, as all resources and agents are assigned to each other. Agents move smartly to locations where they will be needed. We note that the bad agent distribution in the morning (at around) 10:30 am using our approach is an artifact resulting from the initial distribution of agents to uniformly random locations.

V. Competitive Spatiotemporal Search (CSTS) Algorithm

To minimize system-wide average search times, we propose an algorithm to guide drivers that entails two main components. First, we leverage a non-negative matrix factorization approach to build a model that predicts future resource times and locations given a large collection of past trip records. We then leverage this prediction to guide searching agents to minimize their search time, by using a probabilistic search strategy to send a driver to a nearby location where resources are predicted within a short time span. The source code is publicly available at our git repository *https://github.com/joonseok-kim/CompetitiveSearch/*.

A. Spatiotemporal Resource Prediction

Given a large set of resources, each annotated with an origin location, a destination location, and a time-stamp, we first aggregate all information in a spatiotemporal resource origin matrix M. Each cell M_{ij} of M corresponds to the number of resources that became available in a spatial region i during time interval j. We define this matrix formally as follows:

Definition 6 (Spatiotemporal Resource Origin Matrix): Let $\mathcal{D}$ be a training set of resources such that each resource $r_i \in \mathcal{D}$ is a triple (o_i, d_i, t_i), where o_i is the origin of the resource, d_i is the destination of the resource, and t_i is the time at which this resource appears. Further, let $\mathcal{S} = \{S_1, ..., S_{|\mathcal{S}|}\}$ be a set of spatial regions, and let $\mathcal{T} = \{T_1, ..., T_{|\mathcal{T}|}\}$ be a set of temporal intervals spanning all days of $\mathcal{D}$. Then, we define the following $|\mathcal{S}| \times |\mathcal{T}|$ matrix as follows:

$$m_{ij} = |\{r_k \in \mathcal{D} | o_k \in S_i \wedge t_k \in T_j\}|$$

To divide our dataset into spatial regions, we treat each individual road segment as a spatial region and we split time into 20-minute intervals. Thus, we define $\mathcal{S}$ as the set of road segments, and $\mathcal{T}$ as the set of 20-minute intervals of the dataset. We choose 20-minute intervals following our empirical evaluation as described in Section VI. Figure 2 shows the resulting spatiotemporal resource matrix M for the 30 days of June 2016. Each line of this matrix corresponds to an edge in the spatial network $\mathcal{G}$ and each column corresponds

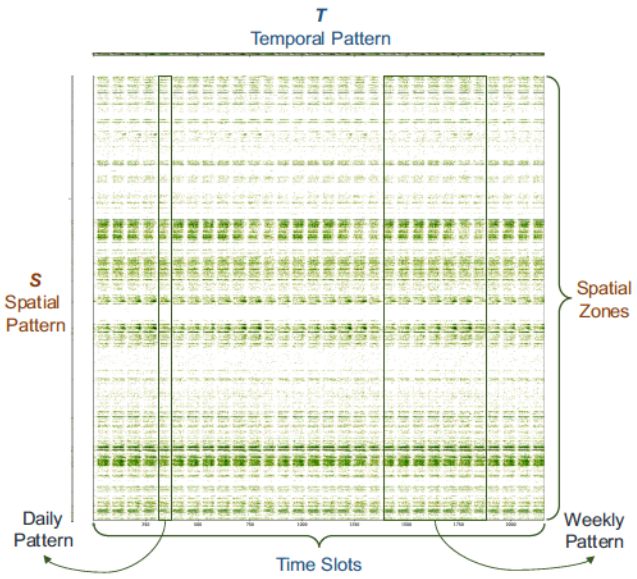


Fig. 2: Spatiotemporal resource matrix

to one of the $30 * 3 * 24 = 2160$ 20-minute intervals during these 30 days. The color intensity of each cell indicates the number of resources observed in the New York City Trip Record Data [1] dataset. We can observe that some edges of the network (lines of the matrix) have little to no resource, while edges have constant high traffic. We can also observe temporal periodic patterns, such as a low number of resources at night, and large numbers during rush hours. We also observe weekly periodic patterns, showing that some edges seem to periodically have less resources on weekends.

To reduce the noise in this matrix and to avoid overfitting, we factorize M using canonical decomposition [9] into two matrices A and B, where A is a $|\mathcal{S}| \times k$ matrix that describes each region in $\mathcal{S}$ by k latent features, and B is a $k \times |\mathcal{T}|$ matrix that describes each time interval by a set of k latent features. Here, k is a parameter that allows us to specify the level of detail of the model. This parameter is selected empirically as $k = 6$ (for this data set) as described in Section VI. Maps showing the distribution of these latent features throughout the road network can also be found in Section VI to help explain the semantic of these features.

Expansion of matrices A and B yields the estimated matrix $\hat{M} = A \cdot B$ that we use for prediction of future resources as follows. For a new day d at a time t, we find the most similar day $\hat{d}$ (using cosine distance) among all days in the training set using the time observed on day d so far. Then we look up the column $\hat{t}$ of $\hat{M}$ that corresponds to time t on day d. The subsequent columns (which corresponds to the resources estimations of the 20-minute intervals after $\hat{t}$) are used for prediction. To describe this approach intuitively, assume it is 8:20am on Sunday s and we want to predict future resources. We use the 25 $(8 * 3 + 1)$ 20-minute intervals seen so far to search for the most similar day in $\hat{M}$. This yields a day $\hat{s}$. To predict today's resource distribution at 8:40 am, we simply use the resource distribution that we have observed on $\hat{s}$ in

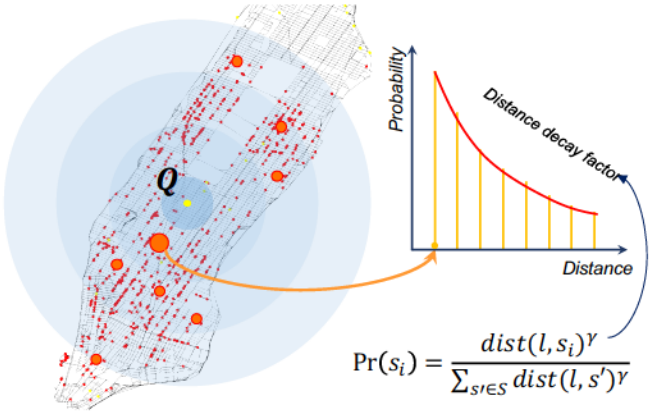


Fig. 3: Sampling and local searching

our training data. We note that our model does not utilize the destination locations of resources, as we see no way to leverage the destination distribution to reduce search times. While it is possible that search times can be reduced by having agents reject resources that lead them to remote places, the simulated setting that we aim to optimize does not allow agents to reject resources.

B. Local Search Guidance

The predicted spatiotemporal resource distribution is used to guide searching agents to minimize their search time. In a nutshell, we first leverage the prediction matrix $\hat{M}$ to draw random candidate resources predicted to appear within a time horizon Δ. To ensure that agents prefer closer locations, we weigh the resulting candidate resources by their distance to the agent. Then, we draw resources from this weighted distribution as the next destination for the agent.

More formally, let t denote the current time, and let $\hat{M}$ be the prediction matrix that predicts the number of resources for each road segment at future time intervals as described in Section V-A. Thus, a cell $\hat{m}_{ij}$ contains the predicted number of resources at road segment S_i at time T_j. Given current location of an agent l, time horizon Δ, sampling size N, and distance decay factor γ, we determine the destination of a searching agent as follows. First, we draw a sample $\mathcal{S}$ of N candidate resources from the distribution of resources estimated within the next Δ minutes:

$$\mathcal{S} = Sample_N(\hat{m}_{\cdot,[t,t+\Delta]}), \tag{1}$$

where $\hat{m}_{\cdot,[t,t+\Delta]}$ is a set of resource predictions of all locations in the network at any times between the current time t and the time horizon $t+\Delta$. The function $Sample_N(\hat{m}_{\cdot,[t,t+\Delta]})$ draws N random sample from the sampling space $\hat{m}_{\cdot,[t,t+\Delta]}$, where each element $\hat{m}_{i,j} \in \hat{m}_{\cdot,[t,t+\Delta]}$ of the sampling space has a probability of $\frac{\hat{m}_{i,j}}{\sum \hat{m}_{\cdot,[t,t+\Delta]}}$ of being drawn.

The resulting set $\mathcal{S}$ contains a set of N candidate locations chosen randomly from the distribution of resources predicted in the next Δ minutes. From this set $\mathcal{S}$ we draw a destination for a searching agent weighted by the distance to the agent's current location l:

$$\mathcal{D} = Sample_1([(s, dist(l,s)^\gamma) | s \in \mathcal{S}]), \tag{2}$$

where $dist(l,s)$ is the network distance between the agent's current location l and a candidate resource $s \in \mathcal{S}$, γ is a parameter that controls the agent's preference to visit nearby locations, $[(s, dist(l,s)^\gamma) | s \in \mathcal{S}]$ is a list (note that this list is not a set, as it may contain duplicates) of pairs of candidate resources in $\mathcal{S}$ and their corresponding distance values taken to the power of γ, and the function $Sample_1$ is the same function that randomly draws a weighted sample from $\mathcal{S}$. That is, each sample $s \in \mathcal{S}$ is chosen with a probability of $\frac{dist(l,s)^\gamma}{\sum_{s' \in \mathcal{S}} dist(l,s')^\gamma}$. The location of this resource D is chosen as destination of the searching agent. This approach of distance-weighted sampling is illustrated in Figure 3 which exemplary shows the search process for the agent Q located at the center of the concentric circles. Small red dots denote all resources predicted to appear within the time horizon Δ. From all these resources, $N = 8$ samples are chosen uniformly at random following Equation 1 highlighted by larger orange dots. Weighted inversely by distance following Equation 2, one of the N samples is chosen as the destination for Q. In this case, as most likely but not certain, the location closest to Q may be chosen, and Q will take the shortest path to this location until assigned a resource.

For our experiments, we choose $\gamma = 0.5$, and $N = 5$ as these values has empirically shown the best reduction in search time as detailed in Section VI.

C. Home Zone Guidance

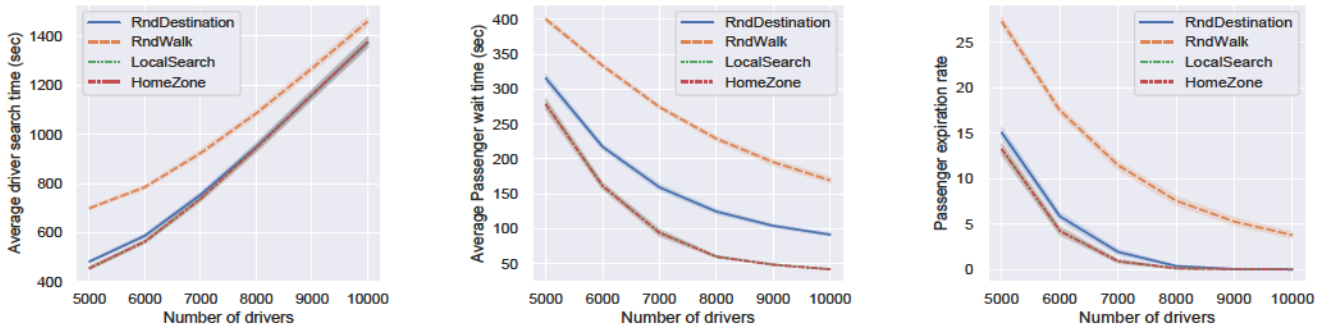
To avoid excessive competition between drivers in busy regions, we map each driver to a dedicated area, denoted as their *home zone*. Similar to local search guidance, home zone guidance relies on a predicted spatiotemporal resource distribution. For allocating resources we now also consider the direction towards the drivers home location. Therefore, customers directly on route to this home zone will get a larger weight versus customers that require the driver to deviate from a respective route. Using the HomeZone heuristic, the probabilistic weight function of Equation 2 is updated as follows:

$$\mathcal{D} = Sample_1([(s, dist(l,s)^\gamma) \cdot (1 - \frac{\sphericalangle(l,s,h)}{\pi}) | s \in \mathcal{S}]), \tag{3}$$

where h is the centroid of the home zone of the driver, and $\sphericalangle(l,s,h) \in [0,\pi]$ denotes the radian angle between line segment $\overleftrightarrow{ls}$ (connecting current driver l and the customer location s) and $\overleftrightarrow{lh}$ (connecting current driver l and home location h).

For example, in case of a customer location being orthogonal to the drivers path towards the home location, the angle would be $90° = 0.5\pi$ yielding a weight of $1 - 0.5 = 0.5$.

The key idea is that drivers will prefer moving towards a busy area nearby their home zone to prevent "herding", and



(a) Average Driver Search Time (b) Average Passenger Wait Time (c) Passenger Expiration Rate

Fig. 4: Experimental Comparison to Baseline Solutions

having agents prefer different areas depending on their home zones. To define home zones of drivers, we subdivide using a regular grid and assign drivers to a grid cell/home zone using a hashing function of the driver ID.

VI. Experimental Evaluation

In this section we provide results of our empirical analysis leveraging the COMSET simulation framework described in Section III using six months of real-world of New York City taxi trip record data [1] entailing a total of 57,697,545 trips. Per default, our simulation uses 5000 agents. To predict resources, we use six-fold cross-validation using, in each fold, one month for testing and the remaining five months for training. Our approach uses a sample size of $N = 5$, a time horizon of $\Delta = 150$ minutes, and a number of latent features $k = 6$ by default, unless noted otherwise.

A. Competitive Spatiotemporal Search Strategy Evaluation

In this section we experimentally evaluate the effectiveness of the distance aware search heuristic (denoted as Local Search) described in Section V-B and the effectiveness of additionally using the Home Zone Guidance heuristic described in Section V-C (denoted as HomeZone). We compare our search heuristics to two baseline strategies:

The *Random Walk* baseline lets agents cruise in random directions. Thus, whenever a random walk agent reaches an intersection, they will continue into a direction chosen uniformly at random. This random walk continues until the agent is assigned a resource. Once a resource is assigned, the random walk is immediately stopped and the agent proceeds to pick up the resource.

The *Random Destination* baseline chooses a random destination uniformly random from all network nodes and follows the shortest path to this destination. Once a resource is assigned to the agent, the shortest path is aborted.

For different numbers of agents, the results of comparing Local Search and HomeZone to these baselines are shown in Figure 4. First, we observe in Figure 4a that the Random Walk approach yields a much higher average search time for drivers than all other approaches. This is due to most agents randomly walking away from the center area where most customers appear. Drivers may stray and stay far from the center, thus becoming unable to help supply the high demand in the center. We also see that, using the average search time as a metric, our Local Search approach yields only a marginal improvement over the random destination approach, especially for a large number of agents. To explain this result, we recall our observations from Section IV: during most times of the simulation, no improvement is possible, as the system is either oversupplied with agents (assigning all resources immediately) or undersupplied (having no agents searching). Thus, the improvement observed in Figure 4a is attributed only to short periods where agents have to compete for resources, leading to a very noticeable improvement during these short times only. Having a larger number of agents leads to permanent oversupply, even during rush hours. In this case, Local Search and Random Destination have most resources instantly assigned, with waiting times only in parts of town where new customers rarely appear. Also, we observe that for the use-case of Manhattan, Random Destination has an advantage: Most of the resources are located in the city center. Agents choosing random destinations are likely to traverse the city center, thus accidental passing through high demand areas. In contrast, Local Search deliberately keeps agents near the center where they are needed, thus reducing search times even with fewer agents. The HomeZone approach yields a slight improvement combining the best of these approaches. While agents choose locations having the highest expected chance for new resources to appear, agent also ensure to choose nearby locations, but also ensure that agents are spatially dispersed to avoid herding by moving towards their home zones.

To better show the difference between these competitors, we turn to another metric to measure the quality of a strategy in Figure 4b. This figure shows that *the average wait time of resources is decreased drastically by both Local Search and HomeZone*. The reason is that our agents are able to anticipate where new resources will appear, thus traveling to these regions ahead of time. In contrast, using Random Destination, agents will have to wait for an agent to randomly pass through. As a consequence, Figure 4c further shows that our approach is able to prevent the expiration of resources, thus

(a) Distance Decay Factor γ (b) Number of NMF features (c) Sample Size

Fig. 5: Parameter Evaluation

allowing to achieve the perfect quality of service for resources at a drastically reduced cost in terms of agents paid and fuel wasted. Next, we explore the effect of the various parameters used by our approach shown in Figure 5.

B. Parameter Evaluation

Distance Decay Factor γ Parameter γ controls how agents prefer close to far-away predicted resources. Figure 5a shows the average search time (and 95% confidence intervals) of 100 simulation runs (with different random seeds) for $-1 \leq \gamma \leq 1$ in 0.1 increments. We first observe that positive values of γ cause more distant places to be weighted higher, thus sending agents to distant resources incurring large search times. We also note that $\gamma < -0.8$ causes agents to become too attracted to their nearest candidate resource, thus getting stuck near their current locations. Empirically, we found that a value of $\gamma = -0.5$, thus using the inverse square root to weigh distances, yields good results. An interesting case is having $\gamma = 0$, where all predicted resources are weighted equally, regardless of their distance. This approach is similar to Random Destination, except that destinations are not chosen uniformly random, but weighted by the predicted resource distribution $\hat{M}$.

Number of Latent Features Figure 5b shows the average agent search time as the number of latent features k for the non-negative matrix factorization in Section V-A is varied. We observe that a value of $k = 6$ is sufficient to yield good performance in terms of search time.

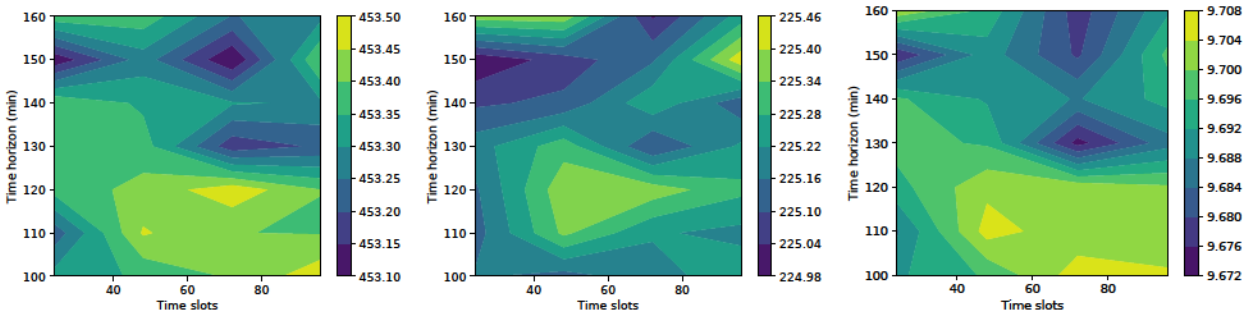
Sampling Size N In Figure 5c, we observe that increasing the number N of candidate resources increases the average search time. The sampling size N defines the number of random resources from which agents chose their destination. Since samples are then selected based on their distance (weighted using parameter γ), a large value of N will lead to shorter distance destinations. Ad-absurdum, for $N = 1$ a random destination is selected within regardless of the distance to the agent. We observe that low values of $N \leq 5$ minimize search times. This indicates that more distance destinations are advisable. However, we note that this behavior may be specific to Manhattan, where long-distance trips lead through the center thus guaranteeing to lead to the main resource hot-spots.

Time Horizon Δ and Temporal Resolution Figure 6 evaluates how our Local Search strategy is affected by different time horizons Δ and different temporal resolution. The x-axis of Figure 6 denotes the number of equal-duration time slots used per day, whereas the $y - axis$ varies the time horizon Δ used to predict future resource availability. We observe in Figure 6a a local minimum in average agent search time having $\Delta = 150min$ and having a day split into 72 intervals ($20min$ each). To further justify our choice of these parameters, as indicated by a little red box, Figures 6b and 6c show that this setting also leads to a low average resource wait time and a low resource expiration rate, respectively.

Number of Home Zones Our experimental evaluation has shown that the number of home zones does not have a significant effect on any of the three metrics average search time, average passenger wait time, and passenger expiration rate. Thus, even as little as two home zones (North and South) are sufficient to ensure that drivers do not congregate in the same location, as half of the agents will drift North (having the Northern Home Zone) and the other half South (having the Southern Home Zone). Increasing the number of different home zones did not show show any significant (beneficial or detrimental) effect such that we omit charts for this parameter.

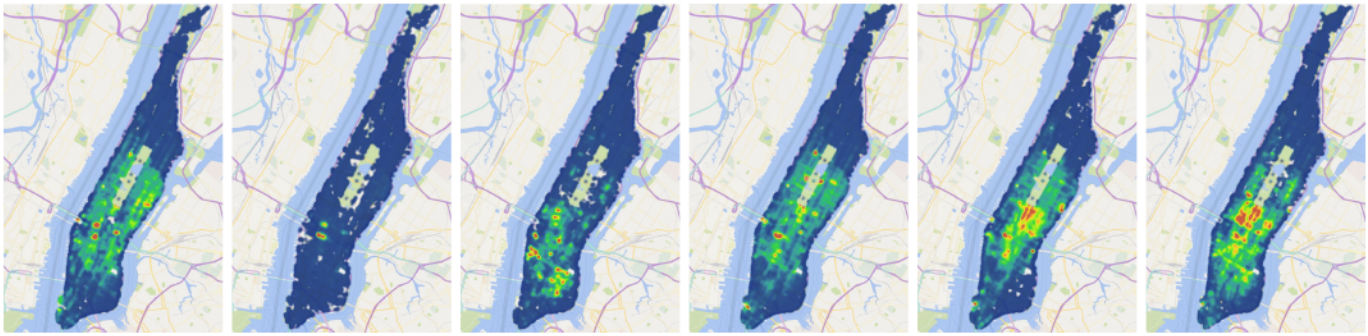
C. Interpretation of Latent Location Features

Towards explainable machine learning, we explore the features learned by the non-negative matrix factorization approach described in Section V-A. Figure 7 visualizes matrix A, the $\mathcal{S} \times k$ matrix which describes each location by $k = 6$ latent features. Figures 7a-f) show the strength of each of these latent features for each edge of the network. An interesting case if Component 2 in Figure 7b), which has high values only in two locations close to each other: Pennsylvania Train Station and Pennsylvania Metro Station, two major transportation hubs.



(a) Average driver search time (b) Average passenger wait time (c) Passenger expiration percentage

Fig. 6: Time Horizon



(a) Component 1 (b) Component 2 (c) Component 3 (d) Component 4 (e) Component 5 (f) Component 6

Fig. 7: Visualization of latent features of matrix A having $k = 6$ latent features

The matrix factorization has learned that these locations have unique temporal features that are not observed in other places. This makes sense, as these locations may have periodical peaks corresponding to the local train and bus schedules. Components 5 and 6 in Figures 7e) and f) have strong features in the center of Manhattan, and may correspond to times when people leave the city, such as the evening rush when most people take taxis to leave the center. Components 1 and 4 in Figures 7a) and d) show a rather uniform distribution, and may indicate features of general demand.

VII. Future Steps Towards Deployment

One inherent weakness of the COMpetitive SEarching Testbed (COMSET) that was used to test and evaluated solutions of competing teams at GIS Cup 2019 is the lack of a traffic model. COMSET uses real-data to adjust the speed of each edge of the road network to reflect its average speed. However, this speed is assumed to be constant regardless of time and traffic conditions. A traffic model is needed to adjust our search heuristic to ensure that the need of customers in slow-traffic regions can be met. Another drawback of COMSET is that, even though the customer distribution follows a real-data distribution, the number of agents is constant due to the lack of available data on ride-hailing drivers. This assumption of a constant number of agents may, however, become realistic in a future of autonomous vehicles that do not need to rest. Furthermore, our setting assumes that the ride-hailing service provider can control the directions of drivers. To deploy our system, incentives would be required for drivers to follow the suggested routes. The home location of a driver should also be considered to prevent a driver from randomly driving far away from their home location, incurring additional overhead to return home. This could be achieved by adding a directional bias to the resource sampling function of Equation 2. It should also be noted that these considerations become less relevant in a future with autonomous cars and continuous operation. Finally, COMSET does not allow multiple passengers to share the same vehicle in a ride-sharing setting, which would be requirement to reduce traffic and the cost to passengers.

VIII. Conclusion

In this work, we tackle the novel problem of competitive spatiotemporal searching for ride-hailing services, which aims at guiding drivers to areas that are likely to have passengers searching for a ride. Applications of this problem not only include better user experience for both drivers and passengers, but also improves traffic conditions by reducing distances traveled by drivers, thus reducing unnecessary emissions and traffic load. Unlike in previous work, our goal is not to find an assignment strategy to match drivers to passengers. Instead, we approach the problem of guiding drivers who cannot currently

be assigned due to a lack of available passengers, and to maximize the chances of finding a passenger in the near future.

To solve this problem, we employ an agent-based simulation approach in which drivers are represented by mobile agents and passengers are represented by immobile resources. Our approach, which is inspired by the results of our visual analysis using taxi-data from New York City, employs a non-negative matrix factorization approach on historic data to predict future resources for the remainder of a new, partially observed day. Using these predicted resources, we use a probabilistic sampling approach to guide agents to predicted locations. We choose a probabilistic approach to prevent agents from greedily herding in the same area, while other areas become undersupplied. Our approach weights sampled locations inversely by distance to avoid extremely long trips while still allowing flexibility to escape unviable regions. Our experimental evaluation shows, qualitatively and quantitatively, that our approach reduces the average search time of drivers, but also reduces the average wait times of passengers. We provide a thorough parameter calibration to obtain the best results for the example of Manhattan.

However, different areas may require different parameter settings, and experimentation for different areas (requiring large sets of trip data in these areas) is required, to assess if the parameter settings for Manhattan generalize to other places. Finally, we discuss simplifying assumptions made by our model, which would need to be addressed for our proposed solution to be deployed by a ride-hailing service provider.

ACKNOWLEDGMENTS

This work was supported by the Defense Advanced Research Projects Agency (DARPA) under cooperative agreement No.HR00111820005 and the National Science Foundation Grant CCF-1637541. The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

REFERENCES

[1] New York City Taxi and Limousine Commission, Trip Record Data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page, Accessed Feb 13, 2020.

[2] N. Agatz, A. L. Erera, M. W. Savelsbergh, and X. Wang. Dynamic ride-sharing: A simulation study in metro atlanta. *Procedia-Social and Behavioral Sciences*, 17:532–550, 2011.

[3] D. Ayala, O. Wolfson, B. Xu, B. Dasgupta, and J. Lin. Parking slot assignment games. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 299–308, 2011.

[4] D. Ayala, O. Wolfson, B. Xu, B. DasGupta, and J. Lin. Spatio-temporal matching algorithms for road networks. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pages 518–521, 2012.

[5] R. Bai, J. Li, J. A. Atkin, and G. Kendall. A novel approach to independent taxi scheduling problem based on stable matching. *Journal of the Operational Research Society*, 65(10):1501–1510, 2014.

[6] D. Bertsimas, P. Jaillet, and S. Martin. Online vehicle routing: The edge of optimization in large-scale applications. *Operations Research*, 67(1):143–162, 2019.

[7] F. Borutta, S. Schmoll, and S. Friedl. Optimizing the spatio-temporal resource search problem with reinforcement learning (gis cup). In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 628–631, 2019.

[8] K. Buchin, I. Kostitsyna, B. Custers, and M. Struijs. A sampling-based strategy for distributing taxis in a road network for occupancy maximization (gis cup). In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 616–619, 2019.

[9] A. Cichocki and A.-H. Phan. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE Trans. Fund. Elect. Comm. Comp. Sci.*, 92(3):708–721, 2009.

[10] G. Cookson. INRIX global traffic scorecard (2017). INRIX Research, February, (published February 2018).

[11] D. Deng, C. Shahabi, U. Demiryurek, and L. Zhu. Task selection in spatial crowdsourcing from worker's perspective. *GeoInformatica*, 20(3):529–568, 2016.

[12] Q. Guo and O. Wolfson. Probabilistic spatio-temporal resource search. *GeoInformatica*, 22(1):75–103, 2018.

[13] Q. Hu, L. Ming, C. Tong, and B. Zheng. An effective partitioning approach for competitive spatial-temporal searching (gis cup). In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 620–623, 2019.

[14] L. Kazemi and C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing. In *Proceedings of the 20th international conference on advances in geographic information systems*, pages 189–198, 2012.

[15] J.-S. Kim, D. Pfoser, and A. Züfle. Distance-aware competitive spatiotemporal searching using spatiotemporal resource matrix factorization (gis cup). In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 624–627, 2019.

[16] G. Li, Y. Zheng, J. Fan, J. Wang, and R. Cheng. Crowdsourced data management: Overview and challenges. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1711–1716, 2017.

[17] D. Moran, K. Kanemoto, M. Jiborn, R. Wood, J. Többen, and K. C. Seto. Carbon footprints of 13 000 cities. *Environmental Research Letters*, 13(6):064041, jun 2018.

[18] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, 2013.

[19] D. Schrank, B. Eisele, T. Lomax, and J. Bak. Urban Mobility Scorecard. The Texas A&M Transportation Institute and INRIX, 2015.

[20] T. Song, Y. Tong, L. Wang, J. She, B. Yao, L. Chen, and K. Xu. Trichromatic online matching in real-time spatial crowdsourcing. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 1009–1020. IEEE, 2017.

[21] A. Tirachini and A. Gomez-Lobo. Does ride-hailing increase or decrease vehicle kilometers traveled (vkt)? a simulation approach for santiago de chile. *International Journal of Sustainable Transportation*, 14(3):187–204, 2020.

[22] Y. Tong, L. Chen, and C. Shahabi. Spatial crowdsourcing: Challenges, techniques, and applications. *Proceedings of the VLDB Endowment*, 10(12):1988–1991, 2017.

[23] Y. Tong, J. She, B. Ding, L. Chen, T. Wo, and K. Xu. Online minimum matching in real-time spatial data: experiments and analysis. *Proceedings of the VLDB Endowment*, 9(12):1053–1064, 2016.

[24] Z. Xu, Z. Li, Q. Guan, D. Zhang, Q. Li, J. Nan, C. Liu, W. Bian, and J. Ye. Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 905–913, 2018.

[25] A. Züfle, G. Trajcevski, D. Pfoser, and J.-S. Kim. Managing uncertainty in evolving geo-spatial data. In *Proceedings of the 21st IEEE International Conference on Mobile Data Management (MDM)*, pages 5–8, 2020.