iLOCuS: Incentivizing Vehicle Mobility to Optimize Sensing Distribution in Crowd Sensing

Susu Xu, Xinlei Chen*, Xidong Pi, Carlee Joe-Wong, Pei Zhang, Hae Young Noh

Abstract—Vehicular crowd sensing systems are designed to achieve large spatio-temporal sensing coverage with low-cost in deployment and maintenance. For example, taxi platforms can be utilized for sensing city-wide air quality. However, the goals of vehicle agents are often inconsistent with the goal of the crowdsourcer. Vehicle agents like taxis prioritize searching for passenger ride requests (defined as task requests), which leads them to gather in busy regions. In contrast, sensing systems often need to sample data over the entire city with a desired distribution (e.g. Uniform distribution, Gaussian Mixture distribution, etc.) to ensure sufficient spatio-temporal information for further analysis. This inconsistency decreases the sensing coverage quality and thus impairs the quality of the collected information. A simple approach to reduce the inconsistency is to greedily incentivize the vehicle agents to different regions. However, incentivization brings challenges, including the heterogeneity of desired target distributions, limited budget to incentivize more vehicle agents, and the high computational complexity of optimizing incentivizing strategies.
To this end, we present a vehicular crowd sensing system to efficiently incentivize the vehicle agents to match the sensing distribution of the sampled data to the desired target distribution with a limited budget. To make the system flexible to various desired target distributions, we formulate the incentivizing problem as a new type of non-linear multiple-choice knapsack problem, with the dissimilarity between the collected data distribution and the desired distribution as the objective function. To utilize the budget efficiently, we design a customized incentive by combining monetary incentives and potential task(ride) requests at the destination. Meanwhile, an efficient optimization algorithm, *iLOCuS*, is presented to plan the incentivizing policy for vehicle agents to decompose the sensing distribution into two distinct levels: time-location level and vehicle level, to approximate the optimal solution iteratively and reduce the dissimilarity objective. Our experimental results based on real-world data show that our system can reduce up to 26.99% of the dissimilarity between the sensed and target distributions compared to benchmark methods.

Index Terms—Mobile Crowd Sensing, Urban Sensing, Connected Vehicles, Sensing Optimization, Incentive Mechanism

1 Introduction

Mobile crowd sensing systems are designed to collect city-wide spatio-temporal data for various purposes, including infrastructure [1], [2], [3], [4], environment [5], [6], [7], [8], social applications [9], [10], [11], etc. Compared to conventional sensing systems, mobile crowd sensing systems can reduce cost and energy consumption by utilizing the low-cost mobile sensors mounted on individual mobile devices [12], [13], [14], [15], [16], [17]. Vehicular crowd sensing systems are a typical example. Mobile sensors are pre-installed on individual vehicles to sense the target data at different time and locations, which reduces the cost to deploy, manage and maintain the mobile sensor system, and becomes more flexible to various short-term tasks [18], [19], [20], [21], [22]. Typically, a vehicular crowd sensing system includes three components: a **data request end**, a **crowdsourcer**, and **vehicle agents**.

The **data request end** requests city-wide sensing data from crowdsourcer for future data analysis. When requesting, the data request end also provides the budget and a desired distribution of the collected data, which we call the target (sensing) distribution. Note we name the distribution/density of the collected data in spatio-temporal domain as the "**sensing distribution**" in the rest of the paper. The target distribution generally depends on the sensing objective of the data request end and consists of the desired information precision in different regions. For example, if the sensing data is collected for general air quality monitoring, the data request end usually expects the collected air pollution data to be uniformly distributed over the city to obtain enough information in different regions for real-time monitoring and forecasting [21], [23], [24]; while when monitoring mobs in a large city, forest fires, factory pollution, or special atmosphere activities during special dates or seasons, the data request end expects to spend the most budget on collecting information in the crowded areas or specific neighborhoods instead of uniformly across the city [25], [26], [27], [28], [29]. With rapidly increasing smart-city applications of vehicular crowdsensing system, the demand of the data request end becomes more diverse, which requires our crowdsourcer to be highly flexible to data requests with various target distributions.

The **vehicle agent** refers to vehicles, e.g. taxis, drones, buses and etc., that have pre-mounted sensors to collect specific types of data at a given sampling frequency while moving through a city. The primary goal of each individual vehicle agent is to finish its original task, e.g. transporting passengers or transporting goods, to make money. Mounting sensors on these non-dedicated vehicles provides a more flexible and cheaper way to collect city-wide data for different application scenarios. For example, taxis with air pollution sensors can monitor city-wide air quality while serving passengers; and delivery drones with cameras can be used for cartography on the way to deliver packages.

The **crowdsourcer** plans sensing data assignments for vehicle

• ** Corresponding author: Xinlei Chen.*
S. Xu, X. Pi and H. Noh are with Department of Civil and Environmental Engineering, Carnegie Mellon University, PA; X. Chen,, C. Joe-Wong, and P. Zhang are with Department of Electrical and Computer Engineering, Carnegie Mellon University, CA. Email: susux@andrew.cmu.edu, xpi@cmu.edu, noh@cmu.edu, xinlei.chen@west.cmu.edu, cjoe-wong@andrew.cmu.edu, peizhang@andrew.cmu.edu.
.

agents and organizes the collected data for the data request end. By integrating all data points collected by all pre-mounted vehicle agents during their movement, the crowdsourcer obtains the final sensing dataset. To satisfy the demand of the data request end, the crowdsourcer needs to 1) optimize the sensing distribution, which ensures the sensing distribution and the target distribution as similar as possible; and 2) be flexible to various desired target distributions of the data request end.

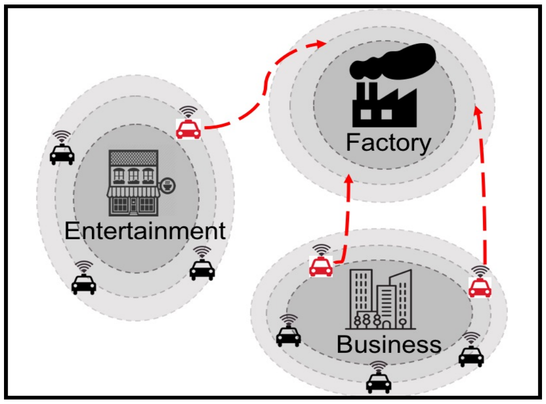


Fig. 1: Incentivizing vehicle agents to achieve uniform distribution over spatial domain.

However, a vehicle agent such as a taxi has a different goal from that of the crowdsourcer. Such inconsistency of goals often results in a lack of data collected in some areas of the city and thus low quality of sensing coverage. The goal of the vehicle agents is to look for more task requests to make money, rather than to sense data, while the crowdsourcer prefers that the taxis distribute themselves according to a target distribution required by the data request end to optimize the sensing quality. For example, to monitor the urban air quality, the data request end needs air pollution data throughout the entire city and across different time intervals to ensure sufficient information for every area [6], [7]. To satisfy the demand of the data request end, the crowdsourcer needs to distribute the air quality measurements uniformly across the city. But taxi drivers spend most time staying in the crowded areas of a city since those areas usually have more ride requests [30], [31]. As a result, few taxis appear in the large non-crowded areas, and the collected air pollution data in these areas is limited. In this case, the sensing system is not able to provide sufficient information about these areas for air pollution monitoring compared to the scenario where taxis distribute uniformly across the city. With the increasing applications of mobile crowdsensing systems, the demand of the data request end may become more and more diverse and not limited to a uniform target distribution [25], [26], [27], [28], [29], [32], [33]. The inconsistency may become more severe when the data request end requests a specifically designed non-uniform distribution. For example, the data request end may request a Gaussian distribution concentrated in factory areas where few taxis pass by. In this case, the difference between the collected data distribution and target distribution is larger than in the case of a uniform target distribution, which will significantly affect the quality of sensing coverage.

A common approach to resolve this problem is to incentivize part of available vehicle agents to new trajectories by offering money, or other forms of non-monetary incentive, e.g. higher probabilities of getting a passenger at the destination, such that the overall distribution of all vehicle agents approximates the target distribution. However, the budget provided by the data request end is often too limited to incentivize all vehicle agents when there is a huge number of vehicle agents.

In this paper, we aim to design a vehicle agent incentivizing algorithm for a crowdsourcer to optimize the sensing distribution and make it close to the target distribution with a limited budget. To optimize the sensing distribution, the key for the crowdsourcer is to figure out 1) which vehicle agents to incentivize, 2) where the vehicle agents should be incentivized to go, and 3) how much to pay for incentivizing each vehicle agent.

However, there are three **challenges** for this objective. 1) For generic target distributions, the difficulty of selecting vehicle agents and their appropriate incentives increases with the complexity of the target distributions. For example, with a uniform target distribution, one can intuitively attempt to ensure equal numbers of vehicle agents in each location. Most previous studies focus on a uniform distribution. When the target distribution dynamically changes over time and space, however, it becomes difficult to decide how to incentivize these vehicle agents. 2) It is difficult to design an incentive that mitigates the inconsistency of goals between the vehicle agents and the crowdsourcer. On the one hand, the crowdsourcer needs to reduce the monetary cost for each vehicle agent to better utilize the budget. On the other hand, the vehicle agents need enough incentives to ensure at least the same profit from following the specified trajectories. 3) There is a large number of vehicle agents, and the number of their candidate trajectories increases exponentially with the length of time, which makes it impossible to use an exhaustive search to obtain the optimal incentive solution.

To address these challenges, our paper introduces a multi-incentive vehicle agent dispatching algorithm. Our algorithm has three major contributions:

- *A novel modeling of the incentivizing problem:* To our best knowledge, we are the first to model the quality of sensing coverage as the *KL-divergence* between the target and sensed data distributions and formulate the sensing coverage optimization problem. We further prove that this formulation is a non-linear multiple-choice knapsack problem, which is NP-complete and impossible to solve in polynomial time.
- *A novel hybrid incentive design to reduce the incentivizing cost*: We design a hybrid incentive for the vehicle agents, which combines the non-monetary incentive of potential task requests at the vehicle agent destination (we call this a "hidden incentive") and the monetary incentive. This combination of incentives allows us to better utilize the budget by decreasing the average cost of incentivizing one vehicle agent.
- *A novel and efficient algorithm to compute optimal the incentivizing strategy*: We introduce the algorithm ***iLOCuS***, which **i**ncentivizes vehic**L**es to **O**ptimize the sensing distribution in a **C**rowd **S**ensing system. The algorithm finds the solution to reduce the dissimilarity in a more efficient way than exhaustive search by a two-stage optimization method.

The rest of the paper is organized as follows: Section 2 introduces related work in optimizing the sensing distribution in mobile sensing networks. Section 3 formulates the problem. Section 4 proposes an optimization algorithm to solve the formulated problem. Section 5 evaluates the proposed problem formulation

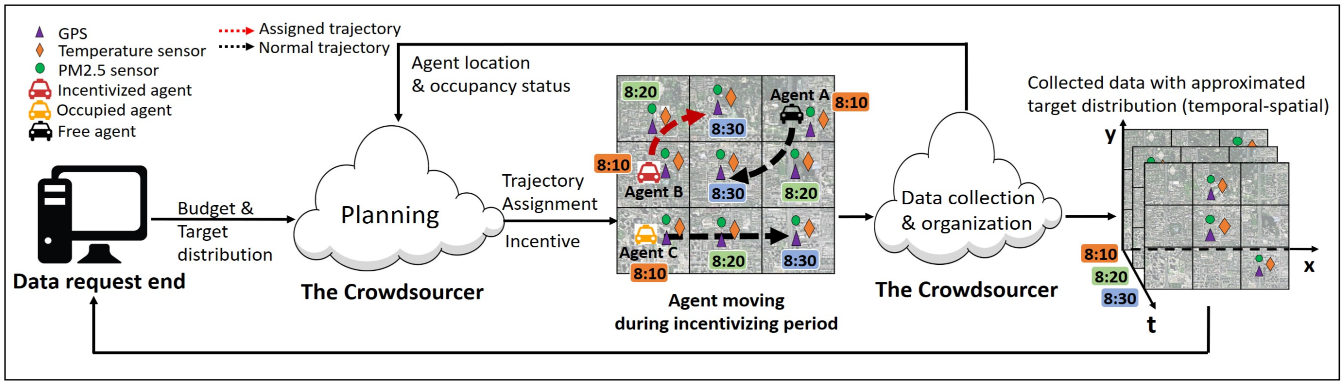


Fig. 2: The diagram of the considered vehicular mobile crowd sensing system.

and algorithm in simulation data. Section 6 summarizes the results and concludes. The Appendix provides the proof of the theorems proposed in the paper.

2 Related Work

We outline the related work on spatio-temporal quality of sensing coverage optimization in mobile sensing networks.

In a mobile sensing system, the quality of sensing coverage refers to a combination of the data coverage, i.e., how many spatial grids and time spots the collected data covers [34], [35], [36], [37], and the data balance level, i.e. whether the collected data has a similar distribution to a given target distribution in the spatio-temporal domain [38], [39]. To achieve a good quality of sensing coverage with a limited budget, most previous works in mobile crowd sensing select a subset of vehicle agents to collect long-term sensory data in their current locations [34], [35], [40], or select dispatching destinations for vehicle agents [35], [39], [41]. These methods mainly focus on static distributions of sensors at some time point, but ignore the influences of the vehicle agents' dynamic mobility on the quality of sensing coverage.

Accounting for the vehicle agents' mobility makes the problem of sensing distribution optimization more complicated. As discussed in [36], the mobility of agents is a double-edged sword. On the one hand, agents' mobility prevents the cost of deploying many fixed sensors to collect data around a large city. On the other hand, the vehicle agents' stochastic and heterogeneous mobility makes it difficult to guarantee a reliable quality of sensing coverage over time. Recently, other works have begun to use the predictable vehicle mobility to improve the vehicle selection in mobile crowd sensing scenario [14], [15], [38], [42], [43]. There are various settings, assumptions, and objectives for these works. For example, [14] discussed how to incentivize vehicles to maximize the total number of covered regions in all time slots or the covered time length in all regions by selecting several vehicles with predicted mobility. [15] minimizes the incentivizing cost considering probabilistic and deterministic mobility models. [25] aims at identifying the important vehicles for the whole network based on their historical mobility patterns, but without considering the quality of spatio-temporal sensing coverage. [38] proposed a framework to optimize the sensing quality in the spatial domain, which assumes the mobility of each user is known and deterministic, and that all users volunteer to sense data without any incentive reward. [43] proposed a reputation-aware framework considering the vehicle availability to select vehicles that achieve target spatial coverage with budget constraints.

Our setup and objective are, in some aspects, different from previous work. In our work, sensors are already pre-mounted on vehicle agents [5], [7], [44] to make it more convenient to collect data, especially for driverless vehicle agents. For the objective, instead of only focusing on spatial coverage or temporal coverage, we aim at the joint spatio-temporal sensing distribution. With this objective, the crowdsourcer better controls the precision of sensing distribution in both time and spatial domains to fulfill the data request end's demand. Meanwhile, instead of directly optimizing the coverage, we make the collected sensing data distribution as similar to the target distribution as possible. In this way, our system is more efficient and flexible to the various demands of data request end on the target distribution. During incentivizing, we not only select part of vehicles from all equipped vehicles, but also decide trajectories for these selected vehicles. This is quite different from previous works which only have vehicle selection but no trajectory selection. Since we jointly optimize the spatio-temporal sensing distribution, every location that the vehicle agents pass by matters in our objective function. On the one hand, being able to select trajectories for some vehicle agents makes it more flexible to incentivize vehicle agents to different locations and achieve better sensing coverage. On the other hand, the trajectory selection makes the problem more computationally complex. This is because we need to select the best trajectory for each vehicle from a huge number of candidate trajectories.

As for the incentive design, many incentivizing mechanisms are proposed based on auction and game-theoretical models, such as reverse auction [45], Stackelberg game [46] and other budget-feasible mechanisms [47], [48], [49], [50]. [51] summarized and compared different types of incentivizing mechanisms for mobile crowdsensing systems. Generally the incentive could be monetary or non-monetary reward. A rule of the thumb of incentive designing is to ensure the value of the incentive is not less than the cost of vehicle agent implementing the sensing assignments.

3 Problem Formulation

Our goal is to incentivize taxi mobility so as to match the collected data distribution to the target distribution with a limited budget and a limited number of vehicle agents. We first define key components of this mobile crowd sensing problem in Section 3.1. Section 3.2 introduces the objective function in detail, which is applicable to various target distributions. Then we describe the design of customized incentives by combining non-monetary rewards to reduce the monetary cost of incentivizing vehicle agents in Section 3.3. Finally, we formulate the problem with physical mobility and budget constraints in Section 3.4.

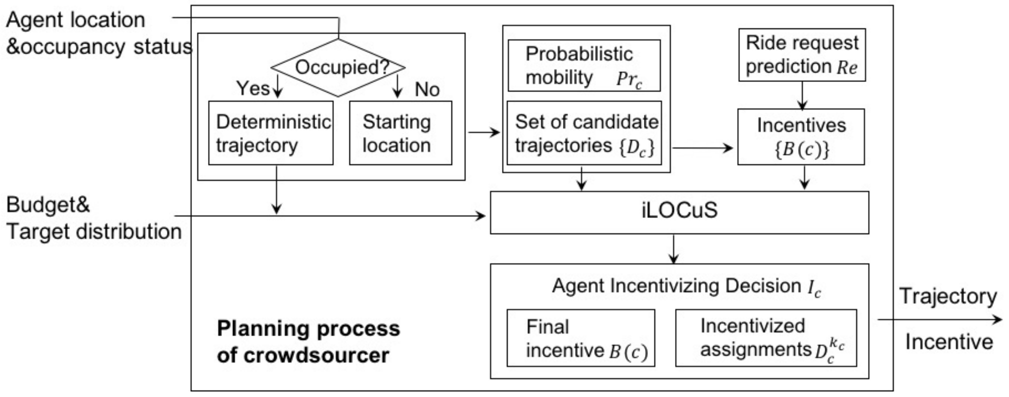

Agent location &occupancy status
Occupied?
Yes
No
Deterministic trajectory
Starting location
Probabilistic mobility Pr_c
Set of candidate trajectories $\{D_c\}$
Ride request prediction Re
Incentives $\{B(c)\}$
Budget& Target distribution
iLOCuS
Agent Incentivizing Decision I_c
Final incentive $B(c)$
Incentivized assignments $D_c^{k_c}$
Planning process of crowdsourcer
Trajectory
Incentive

Fig. 3: The flowchart of crowdsourcer's planning process, which is part of the system described in Figure 2.

3.1 Background and Definitions

In this section, we define key components of our mobile crowd sensing system, as shown in Figure 2. To simplify the problem, the map of a city is discretized into $a \times b$ grids according to the size of the target area and the desired spatial granularity specified by data request end. Time is also discretized into T time slots, where T is the incentivizing period as defined in Section 3.1.1. We denote the grid locations as (i, j), where $1 \leq i \leq a$, $1 \leq j \leq b$, and the current time point is $1 \leq t \leq T$. All vehicle agents are installed with sensors and assumed to run inside the $a \times b$ map region. The data request end provides the budget, target distribution, and other requirements to the crowdsourcer. According to location and occupancy information, the crowdsourcer selects vehicle agents to incentivize and plans trajectories for them. During a specific time period, vehicle agents move inside the target region, either with their usual mobility patterns or following the crowdsourcer's specified trajectory. Data is automatically collected along with the traces of the vehicle agents. The crowdsourcer collects and organizes the uploaded data from all vehicle agents and sends the data to the data request end for further analysis.

3.1.1 *Data request end*

The data request end requests and analyzes the data. The crowd sensing system serves the needs of the data request end. The data request end provides its requirements to the crowdsourcer: the incentivizing period T, the budget B for incentivizing vehicle agents during T, and the target distribution $O(i, j, t)$. Finally, the crowdsourcer returns the crowdsensed data back to the data request end. The crowdsensed data is collected during T without exceeding budget B, and its distribution over time and space is compared to $O(i, j, t)$. If the data request end needs multiple incentivizing periods, it should specify the respective budget and target distributions for each period.

Incentivizing period: denoted as T. At the beginning of each incentivizing period, the crowdsourcer plans and assigns the incentivizing strategies for the next T time points. The length of the incentivizing period indicates how frequently we choose to incentivize a set of vehicles and should be chosen appropriately. If it is too long, the accumulative error of mobility prediction will increase with time and affect our algorithm's performance. If it is too short, it will consume intensive computational resources. If the data request end would like data collected for a longer time span, we can directly incentivize multiple Ts.

Budget: denoted as B, refers to the total amount of money provided by the data request end to incentivize vehicle agents during one incentivizing period.

Target distribution: refers to the desired/expected distribution of data collected over time and space. The target distribution, denoted as O, is a distribution over time and space. $O(i, j, t)$ refers to the percentage of sensing data collected in the location (i, j) at the time point t. Thus, we must have $\sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{t=1}^{T} O(i, j, t) = 1$. The target distribution varies according to the goal of the data analysis. For example, monitoring city-wide air quality requires air pollution data from all regions of the city, and thus requires that the collected data be distributed uniformly over space and time.

3.1.2 *Crowdsourcer*

The crowdsourcer incentivizes vehicle agents based on the provided information from the data request end and current status of each vehicle agent within the incentivizing period. Figure 3 shows the details of the crowdsourcer's planning process. The crowdsourcer takes as input location and occupancy information from the vehicle agents as well as the budget and target distribution from the data request end. The planning process of the crowdsourcer includes three steps: 1) selecting incentivized vehicle agents, where I_c denotes a binary decision to incentivize vehicle agent c, 2) specifying the incentivized trajectory D_c, which will be introduced in Section 3.1.3, for each incentivized vehicle agent c, and 3) designing the customized incentive $B(c)$ to give each incentivized vehicle agent c according to its assignment. After the planning process, if the selected vehicle agents accept the incentive, they move according to the assignment. Meanwhile, no matter whether the vehicle is selected or not, the pre-mounted sensors on the vehicle will automatically collect and upload the sensing data. Finally, the crowdsourcer will organize the collected sensing data from all vehicle agents within an incentivizing period, and send it to the data request end.

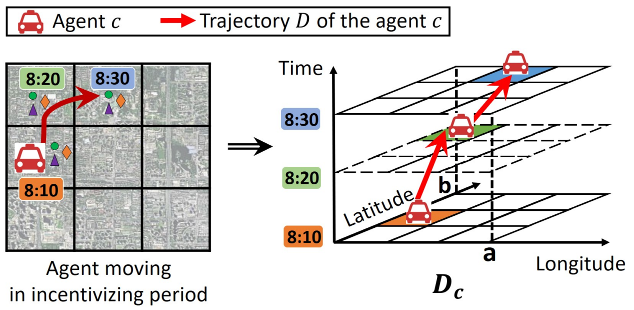

Agent c
Trajectory D of the agent c
8:20
8:30
8:10
Agent moving in incentivizing period
Time
8:30
8:20
8:10
Latitude
b
a
Longitude
D_c

Fig. 4: The 3-D matrix D_c, which represents the trajectory D of vehicle agent c.

3.1.3 Vehicle agent

Vehicle agent refers to an individual vehicle with sensors pre-mounted. The sensors generally include a GPS sensor, an occupancy sensor, and sensors collecting requested information (e.g., air pollutant sensors). The GPS sensor collects location information about the vehicle agent at each time point. The occupancy sensor shows whether the vehicle agent is unoccupied so that crowdsourcer can incentivize it. Since the sensors are pre-mounted on the vehicle agents, it is reasonable to assume that there is no new vehicle entering the system in the incentivizing period.

Status of the vehicle agent: At the beginning of each incentivizing period, all vehicle agents have two possible statuses: unoccupied or occupied. If a vehicle agent is completing an original task at a given time, we mark its status as "occupied" and not available for further incentivizing in its occupied duration. Otherwise, we mark the status of the vehicle agent as "unoccupied". After the incentivizing assignments are distributed, vehicle agents have three possible statuses: free, incentivized or occupied. "Incentivized" vehicle agents follow the incentivized trajectories distributed by the crowdsourcer. "Free" vehicle agents are neither occupied nor incentivized, and cruise according to their usual mobility patterns. Note that since sensors are already installed on the vehicle agents, information is still being collected when the vehicle agent is "occupied" or "free"; however, for an "occupied" or "free" vehicle agent, where data is collected as determined by its original task instead of the crowdsourcer.

Trajectory: denoted as D_c, refers to the mobility of a vehicle agent c during the current incentivizing period. As Figure 4 shows, D_c is a $a \times b \times T$ matrix. Each element of D_c, $0 \leq D_c(i,j,t) \leq 1$, represents the probability that the vehicle agent c appears in (i,j) at time t, where $\sum_{i,j} D_c(i,j,t) = 1$. If $D_c(i,j,t) = 1$, the agent c has deterministic mobility at time t. At the beginning of the incentivizing period, when the crowdsourcer plans the incentivizing strategies, there are two different types of trajectory from the perspective of the crowdsourcer: original trajectory for non-incentivized vehicle agents including occupied and free vehicle agent, incentivized trajectory for incentivized vehicle agents.

- *Original trajectory:* represents the original trajectory of the non-incentivized vehicle agents. Non-incentivized vehicle agents include "occupied" vehicle agents and "free" vehicle agents. The original trajectory of an "occupied" vehicle agent is known to the crowdsourcer, since it is determined by the vehicle agent's original task and reported to crowdsourcer when the task begins. For the "free" vehicle agent which is neither occupied nor incentivized, its original trajectory is probabilistic for the crowdsourcer at the beginning of T. In a general 2-D map grid, there are 9 potential directions that the free vehicle agent can move to in the next time point, which is the adjacent 8 grids and current grid itself (staying in the current location). So the mobility of "free" vehicle agents are stochastic from the perspective of the crowdsourcer. The mobility can be learnt from historical mobility data [52], [53]. So a probabilistic mobility prediction model is employed to predict $Pr_c(i,j,t)$, which refers to the probability that a **free** vehicle agent c appears in the location of (i,j) at time of t. Some empirical mobility prediction models include Markov Chain [54], [55], [56].
- *Incentivized trajectory:* denoted as D_c, refers to the trajectory that crowdsourcer assigns to the incentivized vehicle agent c during T. The assignment is designed by the crowdsourcer so as to achieve the target distribution of the collected data. If a vehicle agent accepts the assignment and respective incentive, it becomes "incentivized" and is not allowed to take tasks during the incentivizing period. The incentivized trajectory of each incentivized vehicle agent c is usually selected from the set of candidate trajectories $\{D_c^{k_c} : k_c \in \{1, \cdots, K_c\}\}$ given K_c deterministic candidate trajectories.

Task request distribution: denoted as $Re(i,j,t)$, refers to the probability that one vehicle agent could obtain at least one task request at a given spatial location (i,j) and time t. It is approximated by the ratio of task request number over the number of unoccupied vehicle agents inside the grid (i,j) at t. The task request probability is marked as 1 if the ratio is higher than 1. Re can be learned and predicted from historical task request data by applying models discussed in [57].

3.2 Objective Function: Dissimilarity between Collected Data Distribution and Target Distribution

The objective of our crowd sensing system is to optimize the sensing distribution such that sufficient information is collected at each time and location. To achieve the objective, in this study, we aim to incentivize the vehicle agents to make the sensing distribution achieve a given target distribution. However, the target distribution may differ according to the demand of data request end. To make the model generalized for heterogeneous target distributions, we define the quality of sensing distribution as how similar the collected data distribution is to the provided target distribution. Therefore, to optimize the quality, we need to reduce the dissimilarity between these two distributions. We measure the dissimilarity using *Kullback-Leibler divergence* from the the collected data distribution to target distribution [57].

In the context of Bayesian inference, the *Kullback-Leibler divergence* from a distribution O to a distribution P, $KL(P||O)$, is a measure of the change of information when one revises beliefs from the prior probability distribution O to the posterior probability distribution P [57]. Optimizing the sensing distribution means minimizing the information changes from the target distribution O to sensing distribution P, which is minimizing the Kullback-Leibler divergence.

In our problem, the target distribution $O(i,j,t)$ over time t and space (i,j) is provided by the data request end. The collected data distribution $P(i,j,t)$ is obtained by integrating all vehicle agents' trajectories in the spatio-temporal domain. Without the loss of generality, we assume all vehicle agents have the same sampling frequency, which is 1 data point per time point per vehicle agent. With all vehicle agents' trajectories $\{D_c : c \in \{1, \cdots, C\}\}$, the amount of collected sensing data at location (i,j) at t is $\sum_{c=1}^{C} D_c(i,j,t)$. The total amount of collected sensing data by all vehicle agents during the whole incentivizing period T is CT. Therefore, given C vehicle agents, the collected sensing data distribution $P(i,j,t)$, which is also the the density of vehicle agents at grid (i,j) at time point t, is calculated as

$$P(i,j,t) = \frac{\sum_{c=1}^{C} D_c(i,j,t)}{CT}.$$

With the target distribution and collected sensing data distribution, the quality of sensing distribution is defined as the negative

TABLE 1: Mathematical Notation

Symbol	**Descriptions of Notations**
$t \in \{1, \cdots, T\}$	tth time unit to collect the data, T time units in one incentivizing period
(i, j)	spatial location where $i \in \{1, \cdots, a\}, j \in \{1, \cdots, b\}$
$c \in \{1, \cdots, C\}$	the cth car in all C cars
P	sensing data distribution over the map grid during the incentivizing period collected by all vehicle agents, with dimension of $a \times b \times T$.
O	target data distribution over the map grid during the incentivizing period, with of $a \times b \times T$.
I_c	a binary indicator showing the vehicle is incentivized or not.
K_c	the number of all deterministic candidate trajectories for vehicle agent c.
k_c	the k_cth trajectory of c, $k_c \in \{0, \cdots, K_c\}$, $k_c = 0$ is the probabilistic trajectory when c cruises without incentivizing or passengers.
D_c	the trajectory of the vehicle agent c, a $a \times b \times T$ matrix, also noted as $D_c^{k_c}$ to distinguish the k_cth candidate trajectories of c.
$B(c)$	expected budget for the cth car to be incentivized to its trajectory D_c.
Re	forecasted task request distribution over time and space in incentivizing period, with dimension of $a \times b \times T$.
Pr_c	mobility prediction of vehicle agent c over time and space in incentivizing period, with the dimension of $a \times b \times T$.
$V(c, D_c^{k_1}, D_c^{k_2})$	the decreasing of *KL-divergence* after switching c from the k_1th to the k_2th trajectory.

of the Kullback-Leibler divergence of P from O:

$$-KL(P||O) = \sum_{i,j,t} P(i,j,t) \log \frac{O(i,j,t)}{P(i,j,t)},$$

Therefore, optimizing the sensing distribution is equivalent to minimizing $KL(P||O)$.

For example, when the target distribution is a uniform distribution, which has probability mass function $O(i, j, t) = const$, the quality of the sensing distribution can be simplified as

$$\begin{aligned} -KL(P||O) &= \sum_{i,j,t} P(i,j,t) \log \frac{const}{P(i,j,t)} \\ &= \log const - \sum_{i,j,t} P(i,j,t) \log P(i,j,t) \end{aligned}$$

The second term $-\sum_{i,j,t} P(i,j,t) \log P(i,j,t)$ is the entropy of the collected data distribution. Previous work [38] utilizes the entropy to evaluate whether the collected data matches the target uniform distribution. Thus, our definition of the sensing distribution quality matches this previous work when the target distribution is uniform.

Furthermore, as opposed to entropy, our objective function, *KL-divergence*, directly measures the dissimilarity between the target distribution and collected sensing data distribution, and thus is more generally applicable to more complex target distributions such as Gaussian mixture distribution, etc.

3.3 Customized Incentives

Our incentivizing system assigns incentives to each vehicle agent to ensure that the vehicle agent is willing to execute the assignments while the total amount of incentives stays within the budget limit. We define the vehicle agent utility as the expected future revenue. Given the incentivizing assignment D_c, the incentive $B(c)$ should cover the possible loss of utility induced by switching from rejecting D_c to accepting D_c. Since the vehicle agents always tend to maximize their utilities, this ensures that they accept the incentives.

Our incentive design's key idea is to incorporate the probability of getting a new task request in the destination of the vehicle agent. Since the primary objective of vehicle agents is to search for potential tasks, if the assigned trajectory brings the vehicle agent to a destination with more tasks compared to the vehicle agent's original trajectory, this improvement is an additional hidden incentive to motivate the vehicle agents accepting the assignment.

We define $r_{\max}$ as the utility from finishing the original task within the incentivizing period of T. The $r_{\max}$ may change over time, since in practice the price of finishing a task may change according to the weather conditions or time of the day. We denote $r_u = r_{\max}/T$ as the utility per time point. We assume r_u and $r_{\max}$ are constant during one incentivizing period T given a short T (e.g. 10 min). Meanwhile, for all vehicle agents, there exists a lower bound $r_{\min}$ for the incentive such that the incentive is not too small to be negligible for vehicle agents. Here we design the incentive $B(c)$ to incentivize the vehicle agent c accepting the assignment D_c as

$$B(c) = \max(r_{\min}, \min(r_{\max}, r_{\max} - r_u(R^c_{ctrl} - R^c_{rand}))). \tag{1}$$

$$R^c_{rand} = \sum_{i,j}^{a,b} Re(i,j,T) Pr_c(i,j,T)$$

$$R^c_{ctrl} = \sum_{i,j}^{a,b} Re(i,j,T) D_c(i,j,T)$$

The task request distribution Re and probabilistic mobility distribution Pr_c are both distributions over the spatio-temporal domain. R^c_{rand} is the expected task request that vehicle agent c could obtain at T by following her/his original trajectory. R^c_{ctrl} is the expected task request that c can obtain in the destination of incentivized trajectory D_c at T.

$B(c)$ is in the range of $[r_{\min}, r_{\max}]$. If the incentivizing assignment D_c helps the vehicle agent c find more task requests, we will take this improvement of task request probability as the hidden incentive and pay less than $r_{\max}$. Otherwise we pay the vehicle agent as much as the utility obtained from original task.

Theorem 1. *$B(c)$ always ensures that utility-maximizing vehicle agents are willing to accept the incentivizing assignment.*

We proved that when incorporating the hidden incentive, the overall utility of the vehicle agent c accepting the incentivizing assignment is larger than the utility of the vehicle agent c rejecting the assignment and running by herself/himself. The proof of Theorem 1 is shown in Appendix A. We also note that $B(c)$ is the minimum incentive that satisfies this property; thus, it is the minimum incentive we can offer rational vehicle agents while ensuring that they will accept.

3.4 Putting It Together: Formulation Of The Vehicle Incentivizing Problem In Crowd Sensing Systems

We formalize our incentivizing policy as the solution to a minimization problem by considering each vehicle agent's physical mobility constraints and the overall budget constraint. For $c \in$

$\{1, \cdots, C\}, t \in \{1, \cdots, T\}, i \in \{1, \cdots, a\}, j \in \{1, \cdots, b\}$, given the target distribution O and budget limit B, our problem is to decide I_c: whether to incentivize vehicles from C, and k_c: the k_cth candidate trajectory selected to be assigned to incentivized vehicle agent c for sensing data collection, such that

$$\min_{\substack{I_1, \cdots, I_C \\ k_1, \cdots, k_C}} KL(P||O) \tag{2}$$

$$\text{subject to } P(i,j,t) = \frac{\sum_{c=1}^{C} D_c(i,j,t)}{CT} \tag{3}$$

$$\sum_{c=1}^{C} B(c) \cdot I_c \leq B \tag{4}$$

$$D_c(i,j,t) \cdot I_c \in \{0, 1\}\ . \tag{5}$$

$$D_c = D_c^{k_c} \text{ where } k_c \in \{0, 1, \cdots, K_c\} \tag{6}$$

Note that $B(c)$ is determined by I_c and k_c and thus is not included as the optimization variable. As Table 1 shows, given the whole map of a target area with longitude of a, latitude of b, and assignment time length T, I_c is a binary indicator of whether the vehicle c is incentivized, and k_c specifies the trajectory assigned to vehicle c, $k_c = 0$ is the probabilistic trajectory when c cruises without incentivizing or passengers, and $k_c > 0$ represents a deterministic incentivized trajectory.

In Section 3.2, we established that optimizing the sensing data distribution is equivalent to minimizing the *KL-divergence*. The target distribution is provided as O. The resulting collected sensing data distribution P should have minimal divergence from O, which is the objective function as Equation 2.

The constraints of this problem include budget constraints (Equation 4) and physical mobility constraints (Equations 5, 6). The budget constraint ensures that the total incentive assigned to all vehicle agents should not exceed the specified budget limit B. Section 3.3 shows how to calculate the incentive $B(c)$ to each vehicle c in Equation 1. To calculate the incentive, the predicted task request distribution Re and predicted vehicle agent's mobility Pr_c depend on the specific application scenario. For example, in a taxi-based sensing platform in which vehicle agents wish to obtain more ride requests, we utilize the model proposed by [58] to forecast the distribution of ride requests. The mobility prediction model of the taxis can also be learned from historical trajectories of vehicles, as in [54]. We discuss the impact of errors in the task request and mobility prediction models in Section 5.

The physical constraints are generated from vehicle agents' mobility. Due to limits on vehicle velocity, each vehicle can either move to a neighboring grid or stay in the original grid within one time unit; it cannot move further. If the vehicle agent c is selected to be incentivized, that is, $I_c = 1$, the incentivized trajectory D_c is specified by the solution, which requires that each element in the matrix of D_c be either 0 or 1. $D_c(i,j,t) \cdot I_c = 1$ means that in the solution, the vehicle agent c is incentivized to pass through the location (i,j) at the time point t. $D_c(i,j,t) \cdot I_c = 0$ means that the incentivized vehicle agent should not pass through the grid (i,j) at t if $I_c = 1$, or the vehicle agent c is not incentivized if $I_c = 0$. Meanwhile, whether vehicle agents are incentivized or not, there is always $\sum_{i,j}^{a,b} D_c(i,j,t) = 1$ for all t. Since each vehicle agent c can only collect one sensing point at any given time point t, the summation of probability that a vehicle agent c appears in different locations at t should be one.

In the next section, an optimization algorithm is introduced to solve the above problem.

4 Algorithm

In this section, we propose a new optimization algorithm, *iLOCuS*, to efficiently solve the problem stated in Section 3.4. Given budget constraints, and vehicle mobility constraints, the algorithm selects a set of vehicle agents and incentivizing trajectories to minimize the objective function. However, the optimization problem is NP-complete, and thus cannot be solved in polynomial time. In this section, we first characterize the hardness of the formulated problem in Section 4.1. Then we propose an optimization algorithm to solve the formulated problem in Section 4.2. In Section 4.3, we discuss the mathematical insights and complexity of the algorithm.

4.1 Problem Characterization

Lemma 1. *This problem is a non-linear multiple-choice knapsack problem, with a convex non-separable objective function and non-continuous variables.*

We first characterize the problem in Lemma 1. The proof is shown in Appendix B. The problem fits the basic form of a non-linear multiple-choice knapsack problem [59], [60]. The optimization version of classic knapsack problem and quadratic knapsack problem are well-known to be NP-hard. In our nonlinear multiple-choice knapsack problem, the objective function becomes the *KL-divergence* between integrated data distribution and target distribution, which makes the problem even harder than the classic linear knapsack problem.

To show that the optimization version of our problem is NP-hard, we first show the decision version of the problem is NP-complete. The decision version of our problem is: *Does there exist a vehicle incentivizing solution such that the* KL-divergence *between collected sensing data distribution and target distribution is smaller than a specific value h while the constraints are satisfied?*

Theorem 2. *The decision version of our problem is NP-complete.*

We showed the proof of Theorem 2 in Appendix C. In Appendix C, we firstly show the decision version of our problem can be verified in polynomial time, hence the problem is NP. Then to prove the problem is NP-hard, we show that one special case of our decision-version problem is equivalent to the decision version of the classic linear multiple-choice knapsack problem, which is widely-known and already proved to be NP-hard [60], [61], [62]. Since the special case is already NP-hard, the decision version of our problem is NP-hard. Therefore, the decision version of our problem is NP-complete. From Theorem 2, since the decision version of our problem has been proven to be NP-complete, it is reasonable to claim that the optimization version of our problem is NP-hard, which is Corollary 1.

Corollary 1. *The optimization version of our problem is NP-hard.*

Based on Theorem 2 and Corollary 1, we show that the formulated problem is NP-hard, which means it is impossible to find an exact optimal solution in a reasonably short time as the scale of the problem increases. The brute-force algorithm has a complexity of $\mathcal{O}(9^T)$, which is not applicable in real-world scenarios. Greedy algorithms can be employed to obtain a sub-optimal, approximated solution. However, the objective function is non-separable with non-continuous variables, which is said to be "much more difficult to solve than the separable problem" [59]. Therefore, it is important to deal with the non-separable function with non-continuous variables where most of the existing greedy algorithms [59], [63], [64], [65] do not apply.

4.2 Proposed Algorithm: *iLOCuS*

To solve the formulated non-linear knapsack problem, we proposed *iLOCuS*. The basic idea of *iLOCuS* is that, instead of directly estimate the gradient of *KL-divergence* with respect to each vehicle agent and its trajectories, we can decompose the non-separable objective function in two stages, firstly by spatio-temporal grid level and then by vehicle agent level. In detail, *iLOCuS* does the following steps in an iterative way: 1) find the time-location pair with the highest ratio between the number of vehicle agents in current solution and the desired vehicle agents at the respective time and location, and 2) dispatch part of passing vehicle agents in the found time-location pair to different trajectories to decrease the *KL-divergence*.

Algorithm 1: *iLOCuS*

```
Input  : Location and occupancy information for all
         vehicle agents
Output: An improved feasible solution $\mathcal{S}^\star$
1  Initialize a feasible solution $S = \{I_c, D_c^{k^*}\}$ and respective
     incentive $\{B(c)\}$ for all $c \in \{1, \cdots, C\}$;
2  set $\mathcal{S}^\star = \mathcal{S}$;
3  for $i++ \leq MaxIter$ do
4      $S = S^*$;
5      Select $(i^*, j^*, t^*) = arg\max_{i,j,t} P(i,j,t)/O(i,j,t)$;
6      Select vehicle agents where $D_c^{k^*}(i^*, j^*, t^*) > 0$ and
         get the set $tmp\_car$;
7      $c = tmp\_car \rightarrow head$;
8      while $c! = null$ do
9          Get potential trace set $\{D_c^k\}$ of vehicle agent $c$,
             where $k \in \{1, \cdots, K\}$;
10         if Total cost is less than $B$ then
11             Select $k = arg\max_k V(c, D_c^{k^*}, D_c^k)$
12         end
13         $c = c \rightarrow next$
14     end
15     Select $(c', k') = arg\max_{c,k} V(c, D_c^{k^*}, D_c^k)$;
16     if $k' > 0$ then
17         Update $\mathcal{S}^*$ as $I'_c = 1$ and $D_{c'}^{k^*} = D_{c'}^{k'}$;
18         Update respective incentive $B(c')$
19     else
20         $I_{c'} = 0, B(c') = 0$;
21     end
22 end
23 Output $\mathcal{S}^\star$ including $I_c$, $D_c = D_c^{k^\star}$, and respective
     incentive $B(c)$.
```

Algorithm 1 describes the steps to keep improving the quality of sensing distribution. We firstly initialize a feasible solution S under the constraints, which is implemented by randomly selecting the vehicle agents until the budget is spent. The solution S includes the binary indicator I_c and feasible assignment $D_c^{k^*}$ for incentivized vehicle agent c. Then we iteratively update S such that the objective function is minimized. During each iteration, we select the maximum value of $P(i,j,t)/O(i,j,t)$, which increases monotonously with the gradient of *KL-divergence* at $P(i,j,t)$. We can obtain the respective time and location pair

$$(i^*, j^*, t^*) = arg\max_{(i,j,t)} P(i,j,t)/O(i,j,t).$$

We need to adapt the number of vehicle agents in (i^*, j^*, t^*) to best decrease the *KL-divergence*. We also defined $V(c, D_c^{k^*}, D_c^{k'})$ to measure how much the *KL-divergence* will decrease when switching the vehicle agent c from current k^*th trajectory to the new k'th trajectory, where

$$\begin{aligned} &V(c, D_c^{k^*}, D_c^{k'}) \\ &= \sum_{i,j,t} (\log \frac{P(i,j,t)}{O(i,j,t)} + 1)(D_c^{k^*}(i,j,t) - D_c^{k'}(i,j,t)) \end{aligned} \quad (7)$$

The basic idea of designing the $V(c, D_c^{k^*}, D_c^{k'})$ is to use the first-order gradient to approximate the difference induced by switching the trajectory. Take (c, k') as $arg\max_{c,k'} V(c, D^{k^*}, D^{k'})$ from all positive V belonging to vehicle agents in (i^*, j^*, t^*). We can decrease the *KL-divergence* approximately most by updating the trajectory of the vehicle agent c to $D^{k'}$ in the current solution S. To distinguish different candidate trajectory for each vehicle agent, we use D_c^k to express the kth possible trajectory of the vehicle agent c. When an unoccupied vehicle agent is not incentivized, she/he follows the usual trajectory, which is a probabilistic trajectory learned from Pr_c, and denoted as D_c^0.

The time complexity of the algorithm is upper bounded by $\mathcal{O}(CT^4)$, where C is the total number of vehicle agents and T is the length of each incentivizing period. For each vehicle agent, it has $\mathcal{O}(T^2)$ potential destinations. Therefore each vehicle agent can only travel in the graph constructed by the $\mathcal{O}(T^2)$ vertices and the $\mathcal{O}(T^2)$ edges connecting those vertices. Using the *Bellman-Ford* algorithm to find the trajectory with maximum value from all the candidate trajectories will cost $\mathcal{O}(T^2 \cdot T^2) = \mathcal{O}(T^4)$. Therefore in the worst case, the overall time complexity for our algorithm is upper bounded by $\mathcal{O}(CT^4)$.

4.3 Insights Behind *iLOCuS*

Our objective function is non-separable with respect to the variables I_c and D_c, since it cannot be converted to the form $\sum_q f_q(I_c, D_c)$, i.e., a linear combination of a group of functions f_q. Meanwhile, both I_c and D_c are not continuous variables. Therefore, it is difficult to use the conventional gradient descent method with respect to the optimization variables I_c and D_c. However, the objective function is convex with respect to $P(i,j,t)$. To minimize the objective function, we can do gradient descent on each $P(i,j,t)$ until the gradient is near 0, which is similar to the idea of coordinate descent with respect to the variables $\{P(i,j,t) : \forall i,j,t\}$. Given a convex function $f(x_1, \cdots, x_n)$ for which it is difficult to simultaneously perform gradient descent with respect to $X = (x_1, \cdots, x_n)$, coordinate descent performs gradient descent at $x_1, \cdots, x_n$ separately to achieve the optimal solution with a much faster convergence rate [66], [67], [68]. The gradient of the objective function with respect to $P(i,j,t)$ is then

$$\frac{\partial KL(P||O)}{\partial P(i,j,t)} = \log \frac{P(i,j,t)}{O(i,j,t)} + 1$$

To accelerate the gradient descent at $P(i,j,t)$, we need to select the steepest direction, which is $(i^*, j^*, t^*) = arg\max_{i,j,t} \log P(i,j,t)/O(i,j,t)$. Then at each iteration, we keep doing gradient descent on each $P(i^*, j^*, t^*)$ until the objective function converges. To decrease the gradient, we need to remove some of the vehicle agents in (i^*, j^*, t^*). Whichever vehicle agent is removed, the gradient with respect to current (i^*, j^*, t^*) always decreases by the same amount as $\log 1/(CT \cdot O(i^*, j^*, t^*)) + 1$.

However, the change in the objective function induced by switching the trajectory varies with different vehicle agents. When removing one vehicle agent at (i^*, j^*, t^*), there must be some other (i', j', t') in which the number of vehicle agents increases. Different $P(i,j,t)$ are no longer independent with each other when changing I_c and D_c^k, since $P(i,j,t)$ is respect to the overall vehicle agents' distribution, while I_c and D_c^k are respect to each vehicle agent's mobility. Therefore, directly using the conventional coordinate descent method cannot solve our problem. It is important to consider how to change I_c, D_c^k to realize the largest decrease of the objective function while performing gradient descent with respect to $P(i,j,t)$.

It is computationally expensive to directly calculate how much the objective function decreases with switching a trajectory. We approximate this change using the product of the gradient and changes of $P(i,j,t)$. When switching the trajectory of one vehicle agent c, only the numbers of vehicle agents at times and locations related to c's old and new trajectories will change. Therefore, only a few $P(i,j,t)$ changes, and the computational efficiency is largely improved. We denote the decreasing of the objective function due to switching vehicle agent c's trajectory from D^{k^*} to $D^{k'}$ as $V(c, D^{k^*}, D^{k'})$. If $V > 0$, the objective decreases. The larger V is, the more the objective function decreases. We have

$$\begin{aligned}&V(c, D^{k^*}, D^{k'})\\ &\propto \sum_{i,j,t} (\log \frac{P(i,j,t)}{O(i,j,t)} + 1)(D_c^{k^*}(i,j,t) - D_c^{k'}(i,j,t)),\end{aligned}$$

It is easy to compute $V(c, D^{k^*}, D^{k'})$, since the deterministic trajectory D_c^k is a sparse matrix. Also, when computing the gradient at the beginning of each iteration, we have stored the matrix of $\log P(i,j,t)/O(i,j,t) + 1$. Denote $C^\star$ as the set of vehicle agent c where $D_c^{k^*}(i^\star, j^\star, t^\star) > 0$. For $c \in C^\star$, we firstly check whether a potential new trajectory $D_c^{k'}$ satisfies the budget constraints and store the respective incentive for late usage. The number of budget-feasible trajectories changes with the amount of leftover budget in the current solution. It is generally much smaller than the total number of candidate trajectories. For a budget-feasible trajectory D_c^k, we can use the dynamic programming method like Bellman-Ford algorithm to obtain the best trajectory $D^{k'}$ for c to incentivize to. If $k' = 0$, it means the best solution is to let the vehicle agent c' run as usual without incentivizing. Otherwise, we will incentivize the vehicle agent c to the trajectory $D_{c'}^{k'}$ instead of its original trajectory. The algorithm repeats the above steps until the *KL-divergence* converges or the maximum number of iterations is achieved. We will output a final solution $\mathcal{S}^\star$, which contains 1) I_c: whether one vehicle agent is incentivized, 2) D_c, the trajectory the vehicle agent c is assigned, and 3) respective incentives $B(c)$ for all incentivized vehicle agents.

5 Evaluation

To evaluate our algorithm, we used a real mobile crowd sensing system based on taxis to collect real-world historical data of taxis' mobility and conduct experiments to show the performance. In Section 5.2 and 5.3, we evaluate the impact of the number of vehicles, budget, target distribution, incentive mechanism, and mobility prediction model accuracy on the *KL-divergence*. We compare the performance of ***iLOCuS*** with no incentivizing, random incentivizing, random incentivizing with the proposed incentives, and Greedy method to optimize the spatial coverage.

5.1 Experiment Setup

5.1.1 Experimental Dataset

This paper uses real-world taxis' trajectories to evaluate our algorithm design. The dataset includes trips of $20,067$ taxis in one month in the city of Beijing, one of the biggest cities in China. Each record in the dataset contains taxi id, time, location and occupancy status. The location is expressed as longitude and latitude while the occupancy status represents whether the taxi is occupied by one or more customers. Each taxi collects one record every minute whenever it is operating. We also have the respective ride request information. The evaluation area occupies a size of $15km$ by $15km$ and is discretized into a 15×15 map grid, in which each grid has the size of 1×1.

We predict the mobility of each vehicle agent using the method developed by [54] where the spatio-temporal dynamics of taxi mobility are probabilistically encoded using a Markov model to improve the prediction accuracy. As for the task request prediction, we utilize the time-space graphical model proposed by [58] and combine historical ride requests of Beijing taxis to forecast the ride request distribution in Beijing city. By learning the temporal evolution and spatial property of the constructed ride request topological graph, this model has been shown to be accurate on the real-world dataset in [58].

We use the Beijing taxi trajectory dataset during November 2015. We discretize the map with 1km grids. We set the incentivizing period as $T = 10$min, which depends on the average distance of a riding, 5km, and the average velocity of vehicle agents, 30km/h. We set the time resolution as 2min, which is the average time needed to run 1 grid. Due to the 2 USD flag-down fare of Beijing taxi, we adopt $r_u = 2$USD/min, $r_{min} = 2$USD and $r_{max} = 20$USD. The first 3 weeks' data is used for training mobility prediction and ride request prediction models, while the rest of the month is used for testing our method. To simplify the problem, we consider the sampling frequency of each vehicle agent as 1 data point per 2 minutes, which means in each grid, a taxi can collect 1 data point. This can be expanded to a higher sampling frequency, but the sensing distribution will not change. We assume that all vehicle agents stay inside the map where the crowdsourcer aims to sense. In order to consider the temporal variations of taxi density, taxi mobility pattern, and ride request density, we take data from multiple times of a day 0:00 am, 6:00 am, 9:00 am 12:00 pm and 6:00 pm.

5.1.2 Benchmark Methods

We adopt three benchmark methods to compare our algorithm with and to validate our algorithm's ability to improve the sensing distribution quality.

- *No Incentivizing (NA)*: This method does not incentivize taxis nor match ride requests. All the taxis just follow their original trajectories. We can check the performance improvement of our method by comparing with NA.
- *Random Incentivizing (RND)*: This method randomly selects taxis and incentivizes them to the sparsest areas with random trajectories within the given budget. *RND* always offers the maximum monetary incentive $r_{\max}$. By comparing this method with our method, we can check the performance improvement brought by our customized incentive and optimization scheme.
- *Random Incentivizing with Ride Request Prediction (RND_RQ)*: This method randomly incentivizes taxis and

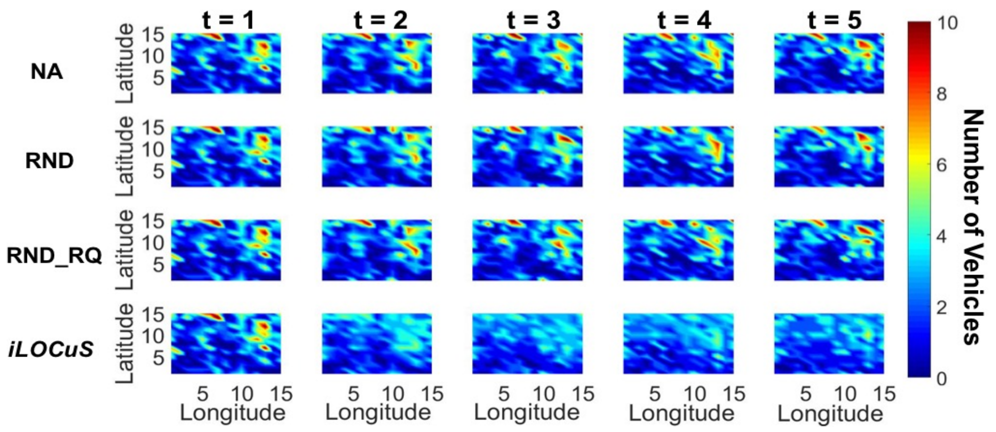


Fig. 5: This figure visualizes the incentivizing results with **Ob_dist_1** as target distribution. The first row is the collected data distribution under no incentivizing, the second row is the distribution under random incentivizing, the third row is the collected data distribution using random incentivizing with proposed incentive, and the fourth row is the incentivizing results under our algorithm *iLOCuS*. The brighter the area is, the denser the vehicles/sensing data points are.

trajectories within the budget while offering the incentive $B(c)$ as described in Equation 1, which includes the ride request in the destination as a hidden incentive. By taking the same incentive mechanism, we compare this method with our method to show the performance improvement of optimizing the incentivizing decision I_c and assignment trajectory D_c.

- *Greedy-SC*: This method is from [14] discussed in Section 2. *Greedy-SC* is developed to greedily choose vehicle agents that can maximize the *Cost Effectiveness* value to maximize the number of covered region in all periods of time using predictable mobility of vehicle agents.

5.1.3 Performance Metrics

We take the objective function, *KL-divergence*, as one of the quantitative evaluation metric. As discussed in Section 3.2, small *KL-divergence* means high quality of sensing distribution. Therefore, the algorithm minimizing *KL-divergence* performs the best.

To show how much the *KL-divergence* is decreased by algorithm $algo^{\star}$ compared to benchmark method $algo$, we define the *divergence reducing percentage (DRP)* as follows:

$$DRP(algo^{\star}, algo) = \frac{KL_{algo} - KL_{algo^{\star}}}{KL_{algo^{\star}}}.$$

DRP measures how much $algo^{\star}$ decreases the *KL-divergence* compared to $algo$, which represents the improvement of the quality of sensing distribution by using the incentivizing policy from $algo^{\star}$. The higher the DRP is, the better the algorithm performs.

5.2 Performance under Multiple Target Distributions

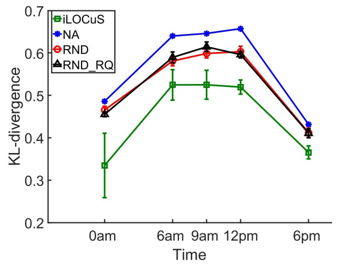
To show the performance under different target distributions, we investigated 4 target distributions under the same budget (1000 USD), vehicle number (500):

- **Ob_dist_1** Uniform distribution over the temporal and spatial domain. For example, when data request end needs the air pollution data over all regions [5], [38].
- **Ob_dist_2** Gaussian distribution over the spatial domain, where the probability mass function achieves the maximum at the grid $(10, 10)$ with fixed variance at each time point. For example, when data request end would like to focus on monitoring the factory neighborhood all day [27], [28], [29].
- **Ob_dist_3** Gaussian mixture distribution over the spatial domain, where the probability mass function is centered at the grid $(5, 10)$ and $(10, 5)$ with fixed variance at each time point. For example, when data request end plans to focus on monitoring both the factory neighborhood and center areas all day [27], [29].
- **Ob_dist_4** Gaussian distribution in spatial domain but the probability mass function achieves the maximum at different grids at different time. For example, when data request end would like to monitor the factory neighborhood during daytime and center area during night [27], [29].

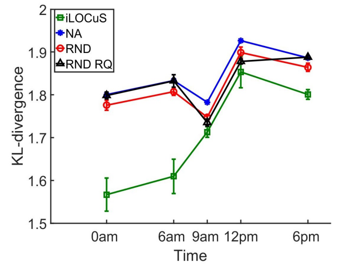
Uniform distribution is one of the most common target distribution for sensing systems [38], [69], [70], which ensures the information is collected uniformly over the temporal and spatial domain. Gaussian distribution and Gaussian mixture distribution over spatial domain consider two common distributions. In these two distributions, the importance level of sensing data changes with their spatial location, while keeping consistent over time. Using these two distributions aims to test the performance of *iLOCuS* in the type of scenario where the data request end may focus more on some specific area(s) and expect to obtain more information near the target spatial area(s). A Gaussian distribution over temporal and spatial domains targets a more complicated distribution, where the importance level of sensing data changes with both time and their spatial location. We investigate the 4 different distributions to show the generalization of our algorithm. Figure 5 visualizes an example of the incentivizing results compared with a target uniform distribution. The sensing distribution, which is also the taxi distribution, at the beginning of one incentivizing period is shown in the column of $t = 1$. While from the next time point $t = 2$ to the end of the incentivizing period, the sensing distribution of *iLOCuS* is much more uniform than the benchmark methods. Figure 5 shows that the dark red and the dark blue areas in the results of no incentivizing (NA), which indicate too much or too few taxis, disappear in the results of *iLOCuS*.

5.2.1 Observations

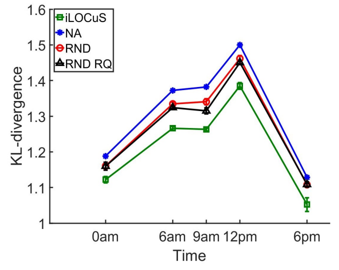
Figure 6 shows that *iLOCuS* always outperforms the benchmark methods, which proves the robustness of *iLOCuS* under different



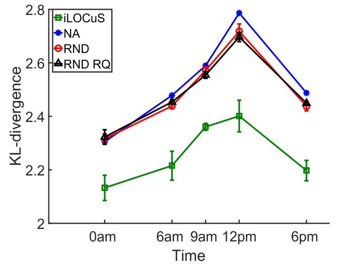
(a) **Ob_dist_1**: Uniform distribution over the temporal and spatial domains



(b) **Ob_dist_2**: Gaussian distribution over the spatial domain



(c) **Ob_dist_3**: Gaussian mixture distribution over the spatial domain



(d) **Ob_dist_4**: Gaussian distribution over space and peak changes with time

Fig. 6: This figure shows the *KL-divergence* under 4 different target distributions at the different time of a day. The figures compare the performance of *iLOCuS* (Square green line) with multiple benchmark methods, including no incentivizing NA (Star blue line), random incentivizing RND (Circle red line) and random incentivizing with the proposed incentive RND_RQ (Triangle black line).

TABLE 2: Divergence reduction percentage of the *KL-divergence* by different algorithms compared to no incentivizing, $DRP(algo^*$, NA), under different target distributions and time of the day. The $algo^*$ includes *iLOCuS*, and benchmark methods (RND and RND_RQ).

Time	**Ob_dist_1** Uniform			**Ob_dist_2** Gaussian over Space			**Ob_dist_3** Gaussian Mixture			**Ob_dist_4** Gaussian over Space and Time		
	iLOCuS	RND	RND_RQ	*iLOCuS*	RND	RND_RQ	*iLOCuS*	RND	RND_RQ	*iLOCuS*	RND	RND_RQ
0:00 am	**45.26%**	4.39%	6.79%	14.92%	1.40%	0.11%	4.66%	0.61%	2.64%	13.12%	0.47%	-0.89%
6:00 am	**22.00%**	10.16%	8.57%	13.89%	1.43%	0.04%	7.26%	1.70 %	3.52%	14.98%	1.89%	0.69%
9:00 am	**23.08%**	7.89%	5.16%	4.06%	2.00%	2.73%	10.48%	4.55%	5.87%	9.12%	-0.60%	0.79%
12:00 pm	**26.52%**	8.99%	10.27%	3.95%	1.46%	2.59%	10.42%	3.42%	3.93%	18.90%	2.27%	4.01%
6:00 pm	**18.09%**	4.72%	5.04%	4.75%	1.19%	-0.09%	5.87%	1.69%	2.07%	18.57%	4.17%	1.81%
Average	**26.99%**	7.23%	7.17%	**8.31**%	1.50%	1.07%	**7.74%**	2.40%	3.61%	**14.94%**	1.64%	1.28%

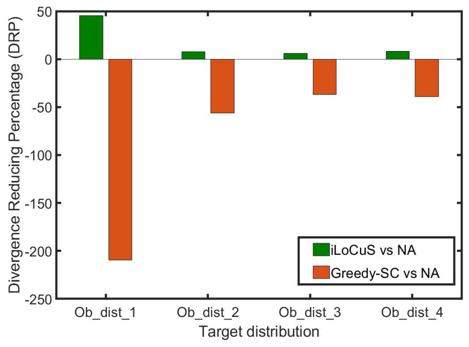


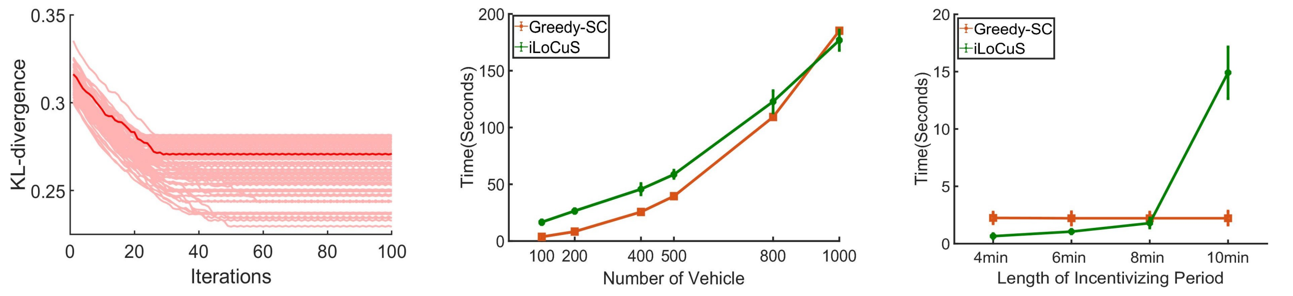
Fig. 7: This figure shows the DRP of *iLOCuS* and baseline method Greedy-SC. It can be found that since Greedy-SC is designed for a different objective function, using Greedy-SC to incentivize vehicle agents will increase the dissimilarity between the collected data distribution and target distribution compared to not incentivizing any vehicle agents.

target distributions. Figure 6(a), 6(b), 6(c) and 6(d) compare the performance of *iLOCuS* with benchmark methods under the **Ob_dist_1**, **Ob_dist_2**, **Ob_dist_3**, and **Ob_dist_4** at different time of a day. The results show that the *KL-divergence* of *iLOCuS* is always smaller than the other benchmark methods under the 4 different target distributions. Table 2 presents the divergence reduction percentage of each method compared to no incentivizing. It is shown that under **Ob_dist_1**, *iLOCuS* reduced the *KL-divergence* of 26.99% on average compared to no incentivizing, which improves 19.76% and 19.82% compared to random incentivizing and random incentivizing with proposed incentive, respectively.

Combining Figure 6 and Table 2, we make several observations that we elaborate below: 1) The *KL-divergence* under no incentivizing (blue line) in Figure 6, mostly achieves the maximum at 12:00 pm and the minimum at 0:00 am or 6:00 pm except **Ob_dist_2**. 2) The *KL-divergence* obtained by our algorithm *iLOCuS* has the similar trend with the *KL-divergence* under no incentivizing. 3) Our algorithm performs best on both *KL-divergence* and DRP under **Ob_dist_1**, which is uniform target distribution. 4) At most time, random incentivizing (random incentivizing and random incentivizing with proposed incentive) performs better than no incentivizing.

For observation (1), at around 12:00 pm, people mostly work and travel in the central area. Thus most taxis also gather in the central area to locate passengers and/or ride requests. This results in a highly concentrated distribution, which is quite different from the target distributions. At 0:00 am, there are few human activities in the central area, and the traffic is less congested. The drivers are able to cruise in a larger area for finding passengers. At 6:00 pm, people head back to the residential areas, which are mostly located on the border of the map. Therefore, at 0:00 am and 6:00 pm, the taxis mostly head to the area which has few taxis at daytime (12:00 pm), which is similar with common commuting pattern in large cities [71]. This makes the sensing distribution less different from our target distributions compared to 12:00 pm. For **Ob_dist_2**, the *KL-divergence* under NA at 9:00 am is similar with 0:00 am. It may be because the taxis gather in the center area closer to the grid of $(10, 10)$ at 9:00 am. The difference between **Ob_dist_2** and **Ob_dist_4** may be because **Ob_dist_4** has changing variance in time domain, while the **Ob_dist_2** requires uniform distribution over the time domain.

For observation (2), our incentivizing algorithm is based on the mobility of all the taxis. There are around $100 - 250$ occupied taxis from the 500 total taxis. These occupied taxis



(a) *KL-divergence* vs Iteration (b) Time vs Number of Vehicle (c) Time vs Length of incentivizing period

Fig. 8: This figure shows the convergence and time complexity with uniform target distribution. (a) shows the *iLOCuS*'s iteration number of convergence with different initialization. (b) shows the computation time of both *Greedy-SC* and *iLOCuS* increases with the number of vehicles and compares the time between *Greedy-SC* and *iLOCuS*. (c) shows the computation time *iLOCuS* increases much faster with the length of the incentivizing period T than *Greedy-SC*. The fast increasing of *iLOCuS*'s time computation is mainly induced by selecting trajectories from exponentially increasing candidate trajectories, while *Greedy-SC* does not allow trajectory selection.

are not able to be incentivized, which means that the quality of sensing distribution will be significantly influenced if these taxis gather in the center area. Meanwhile, if the traffic conditions and weather conditions influence the sensing distribution under no incentivizing, our incentivizing results will be changed in a similar way. Therefore, the performance of *iLOCuS* is closely correlated to the no incentivizing sensing distribution.

For observation (3), compared to the uniform distribution, the three Gaussian distributions, **Ob_dist_2**, **Ob_dist_3** and **Ob_dist_4**, are more concentrated. From Figure 6, we found the *KL-divergence* under NA in uniform distribution ranges in $[0.4, 0.7]$, which is much smaller than the *KL-divergence* under NA in those Gaussian distributions which ranges in $[1.1, 2.8]$. This shows that these Gaussian distributions diverge much more from the real vehicle distribution than uniform distribution. Incentivizing vehicles to the three Gaussian distributions are more difficult than to uniform distribution under the same conditions.

As for observation (4), in general, random incentivizing can make the distribution more uniform than no incentivizing since random incentivizing dispatches the taxis to the destinations where there are fewer number of taxis. But the random incentivizing methods perform much worse than *iLOCuS*, as shown in Table 2. This may be because those random incentivizing methods do not consider the influence of trajectory on our objective function.

We compared *iLOCuS* with *Greedy-SC* as well. Figure 7 compares the DRP between *Greedy-SC* and *iLOCuS* under 4 different target distributions in the same condition. It can be found that directly using *Greedy-SC* will only increase the dissimilarity between the collected data distribution and target distribution. This may be because 1) the objective of *Greedy-SC* only cares about the coverage in the spatial domain, therefore not jointly optimizing the sensing distribution on both temporal and spatial domains; 2) the objective of *Greedy-SC* only cares about the number of covered grids instead of the exact density of collected data points at each grid. So the collected data distribution by *Greedy-SC* is far from the target distribution considering both temporal and spatial domains. The absolute DRP of *Greedy-SC* achieves the maximum for uniform distribution, which may be because the *KL-divergence* is sensitive to the changes of density in all grids in uniform distribution but only center grids in Gaussian distributions.

5.2.2 *Time Complexity*

We also show the convergence of *iLOCuS* with **Ob_dist_1** as the target distribution at different times of the day. Figure 8(a) shows that *iLOCuS* converges after around $20 - 60$ iterations, which shows the convergence. The variance of the converged value of *KL-divergence* is smaller than the variance of *KL-divergence* based on the finally collected data in Figure 6(a). This is because the crowdsourcer optimizes the policy based on the probabilistic mobility prediction of each unoccupied vehicle agents but the vehicle agent will finally only choose one deterministic trajectory during running in real scenarios. By improving the accuracy of mobility prediction, this variation can be mitigated. Figures 8(b) and 8(c) compare the time complexity under different numbers of vehicles and lengths of incentivizing period. The time of *Greedy-SC* increases faster with the number of vehicles than *iLOCuS*, but slower with length of incentivizing period than *iLOCuS*. The complexity of *Greedy-SC* is $\mathcal{O}(C^2T)$ while *iLOCuS* is $\mathcal{O}(CT^4)$. Our *iLOCuS* depends more on the length of incentivizing period is because that *Greedy-SC* only selects vehicle agents but no trajectories while *iLOCuS* selects both vehicle agents and the trajectories for incentivized vehicle agents.

Since *iLOCuS* and *Greedy-SC* have different optimization objective function, here we mainly focusing on comparing their overall performance. In the following exploring influence factors on our *iLOCuS*, we will no longer compare to *Greedy-SC*.

5.3 Influence Factors

In this section, we investigate the performance of *iLOCuS* under different influence factors, including the number of vehicles, budget, incentives, the accuracy of mobility prediction, and vehicle's acceptance rate. We use **Ob_dist_1** as the target distribution for this analysis. The results show that *iLOCuS* is robust to different setups and always outperforms the benchmark methods.

5.3.1 *Number of vehicles*

With uniform target distribution, under different number of vehicles, *iLOCuS* always achieves the lowest *KL-divergence* compared to benchmark methods as Figure 9 shows. With increasing number of vehicles, there are more free vehicles available and thus the *KL-divergence* decreases in all methods. Figure 10 shows a similar trend that the *KL-divergence* decreases with the number of cars with the target distribution of **Ob_dist_2**. For Figure 9(a), 9(b) and red line in Figure 10, we fix the other influence factors as budget and take the average at different times of the day.

We also compare the averaged *KL-divergence* across different times of the day obtained by different algorithms under uniform distribution. We get $DRP(algo, NA)$. $DRP(algo, NA)$ measures how much the *KL-divergence* is reduced by the algorithms

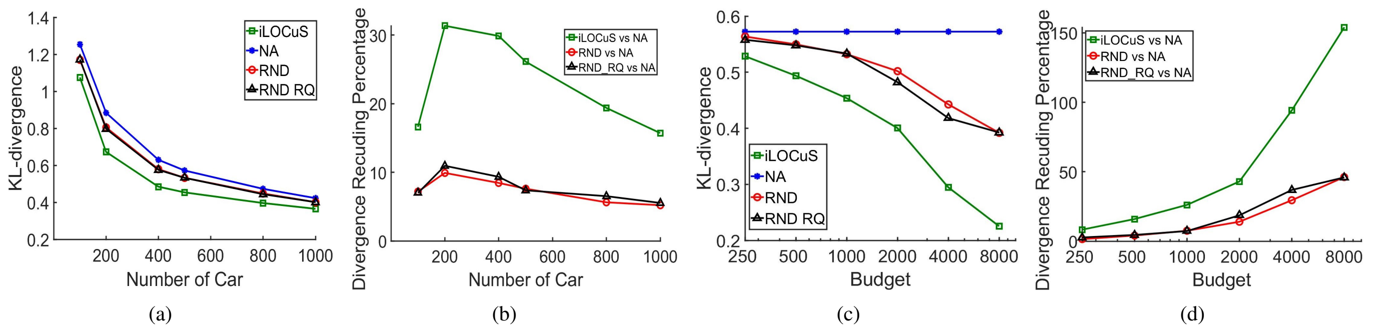


Fig. 9: This figure shows the performance of *iLOCuS* at **Ob_dist_1** target distribution under the different number of vehicle agents and budget. (a) presents how *KL-divergence* changes with increasing number of vehicle agents; (b) shows the DRP comparing the average *KL-divergence* of *iLOCuS*, RND, and RND_RQ over time with the average of NA over time under the different number of vehicle agents. (c) presents how the average *KL-divergence* over different time of the day changes with the amount of budget; (d) shows the DRP comparing the average *KL-divergence* of *iLOCuS*, RND, and RND_RQ over time with the average of NA over time under different budgets.

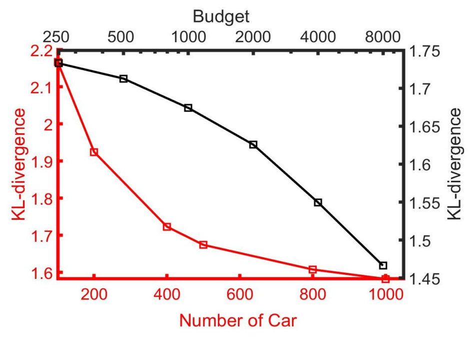


Fig. 10: This figure shows that under **Ob_dist_2**, how the *KL-divergence* of *iLOCuS* changes with number of cars and budget, respectively. The red line and axes represent the changing of *KL-divergence* with number of cars. The black line and axes show the changing of *KL-divergence* with budget.

(*iLOCuS*, *RND*, and *RND_RQ*) compared to no incentivizing (*NA*). It is shown that the DRP of *iLOCuS* outperforms other benchmark methods and achieves upto 31.35% improvement. All algorithms reach their maximums when the number of vehicles is 200. When the number of vehicles is less than 200, more vehicles improve the flexibility for the algorithm to incentivize and incentivize vehicles. So the DRP increases with the number of vehicles increasing to 200, even though the reference *KL-divergence* in no incentivizing keeps decreasing as Figure 9(a) shows. When vehicle number keeps increasing from 200 with the fixed budget, it is possible that when the number of vehicles becomes larger, more vehicles need to be incentivized to realize the same sensing distribution with that under a small number of vehicles. Since all algorithms need to incentivize the vehicles, the budget constraint limits the ability to incentivize more vehicles to more sparse places to improve the sensing quality, which makes the increasing more difficult.

5.3.2 Budget

Figure 9(c) shows that the *KL-divergence* obtained by all methods except no incentivizing decreases with increasing budget with uniform target distribution. Under different available budgets, *iLOCuS* outperforms the benchmark methods. With more budget available, all methods can incentivize more vehicles except no incentivizing. Meanwhile, the *KL-divergence* obtained by *iLOCuS* decreases more quickly than the benchmark methods with budget increasing. This shows that our algorithm utilizes the budget much more efficiently than other methods. Figure 10 shows similar trend that the *KL-divergence* decreases with the budget with the target distribution of **Ob_dist_2**. For Figure 9(c), 9(d) and black line in Figure 10, we fix the number of vehicles as 500 and take an average at the different times of the day.

Meanwhile, as Figure 9(d) shows, DRP of *iLOCuS* increases as budget increases and achieves around 154% when the budget is 8000. The difference between *iLOCuS* and benchmark methods also increases. Compared to no incentivizing and random incentivizing, *iLOCuS* can reduce the cost of incentivizing vehicle agents using ride request as hidden incentives, and realized the same performance (*KL-divergence*) with less budget. Compared to random incentivizing with proposed incentive, *iLOCuS* selects and incentivizes the vehicles which can help reduce the objective value the most, and thus better utilizes the budget.

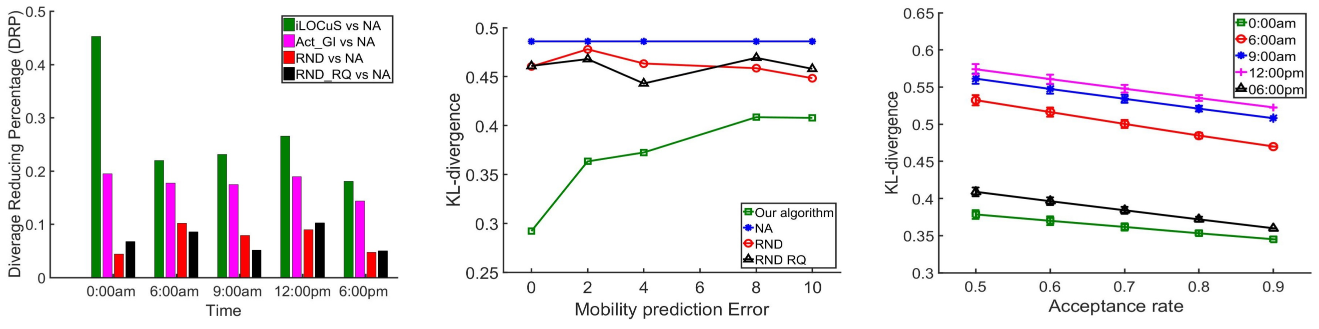
5.3.3 Incentives

We then explore the application of our algorithm under different incentives. We test the performance of our optimization method under fixed incentives $r_{\max}$ instead of $B(c)$ defined in Equation 1. We denote Act_GI as the policy obtained by using our optimization algorithm with all the incentives of $r_{\max}$. We get the divergence reducing percentage of each method compared to no incentivizing. As Figure 11(a) shows, the DRP of all methods rank as *iLOCuS* $> Act\_GI > RND\_RQ \approx RND$, while all the methods outperform than no incentivizing.

The results show 2 points: 1) our optimization algorithm can help decreasing the *KL-divergence* sufficiently with different incentive methods. 2) with our optimization method, the new customized incentive can efficiently improve the performance of our crowd sensing system when comparing *iLOCuS* and Act_GI; We compared the performance of using our optimization method and general constant incentive Act_GI (magenta), with random incentivizing (red) and random incentivizing with proposed incentive (black) in Figure 11(a). Even with constant incentives, our optimization algorithm still reduces the divergence significantly.

5.3.4 Mobility prediction accuracy

We showed the robustness of our algorithm to the different accuracy levels of mobility prediction. Here we need to take mobility prediction results for taxis with different accuracy. When taking mobility distribution with varying bias, we first compute the Euclidean distance between a candidate trajectory to the



(a) (b) (c)

Fig. 11: This figure shows the influence of the incentive design, mobility prediction accuracy, and acceptance rates on the performance of the algorithms. (a) shows the DRP of the *KL-divergence* by using different incentives and different incentivizing algorithms under **Ob_dist_1** at the different time. (b) shows the average *KL-divergence* changes with different mobility prediction bias at 0:00 am. (c) shows the *KL-divergence* under different acceptance rates at the different time in one of our experiments.

trajectory finally chosen by the taxi [72]. Taking the distance d as the variable, we assign probability $p(d)$ obtained from Gaussian distribution $p(d) \sim N(\sigma, \mu^2)$. If the mobility prediction model is accurate, there should be $\sigma = 0$. Here, we set the bias of mobility prediction model ranging in $[0, 10]$. With different configured Gaussian distribution bias, we can obtain multiple mobility prediction models with varying accuracy level. As Figure 11(b) shows, the smaller the error is, the more accurate the mobility prediction model is.

As Figure 11(b) shows, *iLOCuS* is robust to the different accuracy level of mobility prediction model. Although *KL-divergence* increases when the prediction error is larger, *iLOCuS* always outperforms the benchmark methods. The *KL-divergence* gradually increases with increasing prediction error, and converges at around 8. The *KL-divergence* obtained at error of 0 is 71.57% of the *KL-divergence* obtained at error of 10. The increasing pattern of *KL-divergence* is because the mobility prediction influences our estimation on each vehicle's positions in future duration, on which our incentivizing method is based. Thus, when the mobility prediction model has an error, *iLOCuS* may also incentivize vehicles in an inefficient manner.

5.3.5 Vehicle's acceptance rate

Theorem 1 proves that a driver can maximize the monetary utility by accepting the incentivizing assignment. However, in practice, it is possible that some vehicle agents may reject the incentivizing due to emergencies, lack of knowledge, or other preferences. We evaluate the effects of the driver's acceptance rate on the performance of *iLOCuS*. Assuming the acceptance rate r, after getting the incentivizing policy, each vehicle agent has a probability of r to accept the assignments. We conduct the accepting experiment 1000 times given each acceptance rate. Figure 11(c) shows the performance of *iLOCuS* under acceptance rate in the range of $[0.6, 1.0]$ at different times of the day.

As Figure 11(c) shows, the *KL-divergence* obtained at the acceptance rate of 100% ranges between $[88.01\%, 91.17\%]$ of that obtained at the acceptance rate of 60%, which shows the robustness of *iLOCuS* to different acceptance rate. $DRP(\textit{iLOCuS}_1, \textit{iLOCuS}_{0.6})$ ranges in $[9.68\%, 13.62\%]$, where $\textit{iLOCuS}_i$ means the algorithm with acceptance rate of i. The *KL-divergence* decreases linearly with the increasing acceptance rate. All the p-values of the linear regression are less than 0.001, which shows that the linear trends are significant. This figure also shows that if the acceptance rate ranges around $[80\%, 100\%]$, we can ensure at least 93.69% of the performance compared to the 100% acceptance rate using our algorithm *iLOCuS*.

Besides the problem of acceptance, it is also possible that some agent accepts the assignment but does not adhere to the incentivized trajectory. There are two possible solutions to address this adherence problem: 1) if a vehicle agent is subjective malicious, we could exclude it from our vehicle pool. 2) for each candidate trajectory of the malicious vehicle agents, we can adjust the trajectory matrix into a probabilistic format. Based on the adherence history, we can assign a probability to the potential grids that the malicious vehicle agent may deviate from current trajectory to. Our algorithm will automatically evaluate the influence of this deviation to decide whether to incentivize the malicious agents.

6 Conclusion and Future Work

In this paper, we introduce *iLOCuS*, a vehicle incentivizing method to realize a target sensing distribution in mobile crowd sensing system using non-dedicated vehicle platforms. In mobile crowd sensing systems, the inconsistency between the goal of the data request end and the vehicle agents results in the low quality of sensing distribution or requires a large budget. To this end, we formulate the problem into a multiple-choice knapsack problem with a non-separable convex objective function, by considering the budget constraints and physical mobility constraints. We use the task requests at the destination of the incentivizing assignment as a "hidden incentive" to reduce the cost of incentivizing vehicles. Since the formulated problem is NP-complete, we proposed an optimization algorithm to find a solution based on the insight of coordinate descent.

To evaluate the algorithm, we use real Beijing taxi data to show that our system always outperforms benchmark methods under different objective distributions, numbers of vehicles, budgets, lengths of the incentivizing period, incentive mechanisms, mobility prediction accuracy, and incentive acceptance rates. The results show that our algorithm can achieve up to 26.99% improvement in the quality of the sensing distribution compared to not incentivizing the vehicle agents.

In our future work, we plan to better characterize the specific mobility and acceptance rate patterns of each vehicle agent. To better learn the mobility and acceptance rate of taxis, we will use geographical functional zones to discretize the spatial locations instead of directly dividing the map into grids and then use history of acceptance behaviors to predict their future acceptance rate.

REFERENCES

[1] Dragomir Anguelov, Carole Dulong, Daniel Filip, Christian Frueh, Stéphane Lafon, Richard Lyon, Abhijit Ogale, Luc Vincent, and Josh Weaver. Google street view: Capturing the world at street level. *Computer*, 43(6):32–38, 2010.

[2] Xu Li, Wei Shu, Minglu Li, Hong-Yu Huang, Pei-En Luo, and Min-You Wu. Performance evaluation of vehicle-based mobile sensor networks for traffic monitoring. *IEEE transactions on vehicular technology*, 58(4):1647–1653, 2009.

[3] Susu Xu, Lin Zhang, Pei Zhang, and Hae Young Noh. An indirect traffic monitoring approach using building vibration sensing system. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, pages 374–375. ACM, 2016.

[4] Susu Xu, Lin Zhang, Pei Zhang, and Hae Young Noh. An information-theoretic approach for indirect train traffic monitoring using building vibration. *Frontiers in Built Environment*, 3:22, 2017.

[5] Srinivas Devarakonda, Parveen Sevusu, Hongzhang Liu, Ruilin Liu, Liviu Iftode, and Badri Nath. Real-time air quality monitoring through mobile sensing in metropolitan areas. In *Proceedings of the 2nd ACM SIGKDD international workshop on urban computing*, page 15. ACM, 2013.

[6] Yu Zheng, Furui Liu, and Hsun-Ping Hsieh. U-air: When urban air quality inference meets big data. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1436–1444. ACM, 2013.

[7] Xinlei Chen, Xiangxiang Xu, Xinyu Liu, Hae Young Noh, Lin Zhang, and Pei Zhang. Hap: Fine-grained dynamic air pollution map reconstruction by hybrid adaptive particle filter. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, pages 336–337. ACM, 2016.

[8] Yin Wang, Xuemei Liu, Hong Wei, George Forman, Chao Chen, and Yanmin Zhu. Crowdatlas: Self-updating maps for cloud and personal use. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*, pages 27–40. ACM, 2013.

[9] Shane B Eisenman, Emiliano Miluzzo, Nicholas D Lane, Ronald A Peterson, Gahng-Seop Ahn, and Andrew T Campbell. Bikenet: A mobile sensing system for cyclist experience mapping. *ACM Transactions on Sensor Networks (TOSN)*, 6(1):6, 2009.

[10] Xidong Pi, Zhen Qian, Aaron Steinfeld, and Yun Huang. Understanding human perception of bus fullness: An empirical study of crowdsourced fullness ratings and automatic passenger count data. *Transportation Research Record*, page 0361198118781398, 2018.

[11] Xinlei Chen, Yu Wang, Jiayou He, Shijia Pan, Yong Li, and Pei Zhang. Cap: Context-aware app usage prediction with heterogeneous graph embedding. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(1), 2019.

[12] Raghu K Ganti, Fan Ye, and Hui Lei. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, 49(11), 2011.

[13] Zhuojun Duan, Mingyuan Yan, Zhipeng Cai, Xiaoming Wang, Meng Han, and Yingshu Li. Truthful incentive mechanisms for social cost minimization in mobile crowdsourcing systems. *Sensors*, 16(4):481, 2016.

[14] Zongjian He, Jiannong Cao, and Xuefeng Liu. High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility. In *Computer Communications (INFOCOM), 2015 IEEE Conference on*, pages 2542–2550. IEEE, 2015.

[15] Xiumin Wang, Weiwei Wu, and Deyu Qi. Mobility-aware participant recruitment for vehicle-based mobile crowdsensing. *IEEE Transactions on Vehicular Technology*, 67(5):4415–4426, 2018.

[16] Shuguan Yang and Zhen Sean Qian. Turning meter transactions data into occupancy and payment behavioral information for on-street parking. *Transportation Research Part C: Emerging Technologies*, 78:165–182, 2017.

[17] Shuguan Yang and Sean Qian. A non-sensor solution for effective and inexpensive parking management: payment, reservation, and dynamic pricing.

[18] Kashif Ali, Dina Al-Yaseen, Ali Ejaz, Tayyab Javed, and Hossam S Hassanein. Crowdits: Crowdsourcing in intelligent transportation systems. In *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*, pages 3307–3311. IEEE, 2012.

[19] Vitaly Petrov, Andrey Samuylov, Vyacheslav Begishev, Dmitri Moltchanov, Sergey Andreev, Konstantin Samouylov, and Yevgeni Koucheryavy. Vehicle-based relay assistance for opportunistic crowdsensing over narrowband iot (nb-iot). *IEEE Internet of Things Journal*, 2017.

[20] Zhaolong Ning, Feng Xia, Noor Ullah, Xiangjie Kong, and Xiping Hu. Vehicular social networks: Enabling smart mobility. *IEEE Communications Magazine*, 55(5):16–55, 2017.

[21] Elizabeth Bales, Nima Nikzad, Nichole Quick, Celal Ziftci, Kevin Patrick, and William Griswold. Citisense: Mobile air quality sensing for individuals and communities design and deployment of the citisense mobile air-quality system. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on*, pages 155–158. IEEE, 2012.

[22] Jen-Hao Liu, Yu-Fan Chen, Tzu-Shiang Lin, Da-Wei Lai, Tzai-Hung Wen, Chih-Hong Sun, Jehn-Yih Juang, and Joe-Air Jiang. Developed urban air quality monitoring system based on wireless sensor networks. In *Sensing technology (icst), 2011 fifth international conference on*, pages 549–554. IEEE, 2011.

[23] Ricardo Piedrahita, Yun Xiang, Nick Masson, John Ortega, Ashley Collier, Yifei Jiang, Kun Li, Robert P Dick, Qin Lv, Micahel Hannigan, et al. The next generation of low-cost personal air quality sensors for quantitative exposure monitoring. *Atmospheric Measurement Techniques*, 7(10):3325–3336, 2014.

[24] Xinlei Chen, Xiangxiang Xu, Xinyu Liu, Shijia Pan, Jiayou He, Hae Young Noh, Lin Zhang, and Pei Zhang. Pga: Physics guided and adaptive approach for mobile fine-grained air pollution estimation. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, pages 1321–1330. ACM, 2018.

[25] Kavi K Khedo, Rajiv Perseedoss, Avinash Mungur, et al. A wireless sensor network air pollution monitoring system. *arXiv preprint arXiv:1005.1737*, 2010.

[26] Uichin Lee, Biao Zhou, Mario Gerla, Eugenio Magistretti, Paolo Bellavista, and Antonio Corradi. Mobeyes: smart mobs for urban monitoring with a vehicular sensor network. *IEEE Wireless Communications*, 13(5), 2006.

[27] Dimitri Zarzhitsky, Diana F Spears, William M Spears, and David R Thayer. A fluid dynamics approach to multi-robot chemical plume tracing. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1476–1477. IEEE Computer Society, 2004.

[28] Jude Allred, Ahmad Bilal Hasan, Saroch Panichsakul, William Pisano, Peter Gray, Jyh Huang, Richard Han, Dale Lawrence, and Kamran Mohseni. Sensorflock: an airborne wireless sensor network of micro-air vehicles. In *Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 117–129. ACM, 2007.

[29] Eric Paulos, Richard J Honicky, and Elizabeth Goodman. Sensing atmosphere. *Human-Computer Interaction Institute*, page 203, 2007.

[30] Graham Currie. Gap analysis of public transport needs: measuring spatial distribution of public transport needs and identifying gaps in the quality of public transport provision. *Transportation Research Record: Journal of the Transportation Research Board*, (1895):137–146, 2004.

[31] Yu Liu, Chaogui Kang, Song Gao, Yu Xiao, and Yuan Tian. Understanding intra-urban trip patterns from taxi trajectory data. *Journal of geographical systems*, 14(4):463–483, 2012.

[32] Xinlei Chen, Aveek Purohit, Carlos Ruiz Dominguez, Stefano Carpin, and Pei Zhang. Drunkwalk: Collaborative and adaptive planning for navigation of micro-aerial sensor swarms. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 295–308. ACM, 2015.

[33] Xinlei Chen, Aveek Purohit, Shijia Pan, Carlos Ruiz, Jun Han, Zheng Sun, Frank Mokaya, Patric Tague, and Pei Zhang. Design experiences in minimalistic flying sensor node platform through sensorfly. *ACM Transactions on Sensor Networks (TOSN)*, 13(4):33, 2017.

[34] Luis G Jaimes, Idalides Vergara-Laurens, and Miguel A Labrador. A location-based incentive mechanism for participatory sensing systems with budget constraints. In *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, pages 103–108. IEEE, 2012.

[35] Mohamed Younis and Kemal Akkaya. Strategies and techniques for node placement in wireless sensor networks: A survey. *Ad Hoc Networks*, 6(4):621–655, 2008.

[36] Huadong Ma, Dong Zhao, and Peiyan Yuan. Opportunities in mobile crowd sensing. *IEEE Communications Magazine*, 52(8):29–35, 2014.

[37] Xinlei Chen, Susu Xu, Haohao Fu, Carlee Joe-Wong, Lin Zhang, Hae Young Noh, and Pei Zhang. Asc: Actuation system for city-wide crowdsensing with ride-sharing vehicular platform. In *14th International Science of Smart City Operations and Platforms Engineering*. IEEE, 2019.

[38] Shenggong Ji, Yu Zheng, and Tianrui Li. Urban sensing based on human

mobility. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 1040–1051. ACM, 2016.

[39] Zheng Song, Chi Harold Liu, Jie Wu, Jian Ma, and Wendong Wang. Qoi-aware multitask-oriented dynamic participant selection with budget constraints. *IEEE Transactions on Vehicular Technology*, 63(9):4618–4632, 2014.

[40] Yazhi Liu, Jianwei Niu, and Xiting Liu. Comprehensive tempo-spatial data collection in crowd sensing using a heterogeneous sensing vehicle selection method. *Personal and Ubiquitous Computing*, 20(3):397–411, 2016.

[41] Fang-Jing Wu, Yu-Fen Kao, and Yu-Chee Tseng. From wireless sensor networks towards cyber physical systems. *Pervasive and Mobile computing*, 7(4):397–413, 2011.

[42] Junaid Ahmed Khan, Yacine Ghamri-Doudane, and Dmitri Botvich. Autonomous identification and optimal selection of popular smart vehicles for urban sensingan information-centric approach. *IEEE Transactions on Vehicular Technology*, 65(12):9529–9541, 2016.

[43] Sherin Abdelhamid, Hossam S Hassanein, and Glen Takahara. Reputation-aware, trajectory-based recruitment of smart vehicles for public sensing. *IEEE Transactions on Intelligent Transportation Systems*, 19(5):1387–1400, 2018.

[44] Yajie Ma, Mark Richards, Moustafa Ghanem, Yike Guo, and John Hassard. Air pollution monitoring and mining based on sensor grid in london. *Sensors*, 8(6):3601–3623, 2008.

[45] Juong-Sik Lee and Baik Hoh. Sell your experiences: a market mechanism based incentive for participatory sensing. In *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pages 60–68. IEEE, 2010.

[46] Dejun Yang, Guoliang Xue, Xi Fang, and Jian Tang. Incentive mechanisms for crowdsensing: crowdsourcing with smartphones. *IEEE/ACM Transactions on Networking*, 24(3):1732–1744, 2016.

[47] Georgios Amanatidis, Georgios Birmpas, and Evangelos Markakis. *Coverage, Matching, and Beyond: New Results on Budgeted Mechanism Design*. Springer Berlin Heidelberg, 2016.

[48] Zhenzhe Zheng, Fan Wu, Xiaofeng Gao, Hongzi Zhu, Guihai Chen, and Shaojie Tang. A budget feasible incentive mechanism for weighted coverage maximization in mobile crowdsensing. *IEEE Transactions on Mobile Computing*, PP(99):1–1, 2017.

[49] Luis G. Jaimes, Idalides J. Vergara-Laurens, and Andrew Raij. A survey of incentive techniques for mobile crowd sensing. *IEEE Internet of Things Journal*, 2(5):370–380, 2015.

[50] Susu Xu, Weiguang Mao, Yue Cao, Hae Young Noh, and Nihar B Shah. An incentive mechanism for crowd sensing with colluding agents. *arXiv preprint arXiv:1809.05161*, 2018.

[51] Robert Ighodaro Ogie. Adopting incentive mechanisms for large-scale participation in mobile crowdsensing: from literature review to a conceptual framework. *Human-centric Computing and Information Sciences*, 6(1):24, 2016.

[52] Xidong Pi and Zhen Sean Qian. A stochastic optimal control approach for real-time traffic routing considering demand uncertainties and travelers choice heterogeneity. *Transportation Research Part B: Methodological*, 104:710–732, 2017.

[53] Wei Ma and Zhen Sean Qian. Estimating multi-year 24/7 origin-destination demand using high-granular multi-source traffic data. *Transportation Research Part C: Emerging Technologies*, 96:96–121, 2018.

[54] Siyu Chen, Yong Li, Wenyu Ren, Depeng Jin, and Pan Hui. Location prediction for large scale urban vehicular mobility. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*, pages 1733–1737. IEEE, 2013.

[55] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *Robomech Journal*, 1(1):1, 2014.

[56] Brendan Tran Morris and Mohan M Trivedi. Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. *IEEE transactions on pattern analysis and machine intelligence*, 33(11):2287–2301, 2011.

[57] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[58] Abhinav Jauhri, Brian Foo, Jerome Berclaz, Chih Chi Hu, Radek Grzeszczuk, Vasu Parameswaran, and John Paul Shen. Space-time graph modeling of ride requests based on real-world data. *arXiv preprint arXiv:1701.06635*, 2017.

[59] Kurt M Bretthauer and Bala Shetty. The nonlinear knapsack problem–algorithms and applications. *European Journal of Operational Research*, 138(3):459–472, 2002.

[60] Hans Kellerer, Ulrich Pferschy, and David Pisinger. Introduction to np-completeness of knapsack problems. In *Knapsack problems*, pages 483–493. Springer, 2004.

[61] Toshihide Ibaraki. Approximate algorithms for the multiple-choice continuous knapsack problems. *Journal of the Operations Research Society of Japan*, 23(1):28–63, 1980.

[62] Toshihide Ibaraki, Toshiharu Hasegawa, Katsumi Teranaka, and Jiro Iwase. The multiple-choice knapsack problem. *Journal of the Operations Research Society of Japan*, 21(1):59–95, 1978.

[63] Thomas Blumensath and Mike E Davies. Iterative thresholding for sparse approximations. *Journal of Fourier analysis and Applications*, 14(5-6):629–654, 2008.

[64] Pinghua Gong, Changshui Zhang, Zhaosong Lu, Jianhua Huang, and Jieping Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *International Conference on Machine Learning*, pages 37–45, 2013.

[65] Jérôme Bolte, Shoham Sabach, and Marc Teboulle. Proximal alternating linearized minimization or nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1-2):459–494, 2014.

[66] Tong Tong Wu, Kenneth Lange, et al. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, 2(1):224–244, 2008.

[67] Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. Coordinate descent method for large-scale l2-loss linear support vector machines. *Journal of Machine Learning Research*, 9(Jul):1369–1398, 2008.

[68] Yu Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[69] Yu Zheng, Tong Liu, Yilun Wang, Yanmin Zhu, Yanchi Liu, and Eric Chang. Diagnosing new york city's noises with ubiquitous data. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 715–725. ACM, 2014.

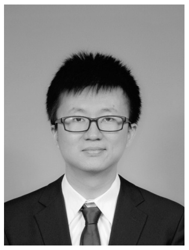
[70] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2267–2276. ACM, 2015.

[71] Pengjun Zhao, Bin Lü, and Gert de Roo. Urban expansion and transportation: the impact of urban form on commuting patterns on the city fringe of beijing. *Environment and Planning A*, 42(10):2467–2486, 2010.

[72] Weiming Hu, Xuejuan Xiao, Zhouyu Fu, Dan Xie, Tieniu Tan, and Steve Maybank. A system for learning statistical motion patterns. *IEEE transactions on pattern analysis and machine intelligence*, 28(9):1450–1464, 2006.

Susu Xu received the B.E. degree from Tsinghua University, China in 2014, and M.S. degree from Carnegie Mellon University, Pittsburgh, PA, USA. She is now pursuing her Ph.D. degree in Civil and Environmental Engineering in Carnegie Mellon University, Pittsburgh, PA, USA. Her research interests are mobile crowd sensing system, incentive mechanism, optimization, and time-series signal processing.

Xinlei Chen received the B.E. and M.S. degrees in electrical engineering from Tsinghua University, China, in 2009 and 2012, respectively, and Ph.D degrees in Electrical Engineering from Carnegie Mellon University, Pittsburgh, PA, USA. His research interests are networking and communications, including millimeter wave communications, device-to-device communication, mobile embedded system, including collaborative drone swarms, and big data.



Xidong Pi received the B.E. degrees from Tsinghua University, China in 2014, and the M.S. degree from Carnegie Mellon University, Pittsburgh, PA, USA. He is now a Ph.D. candidate in Civil and Environmental Engineering in Carnegie Mellon University. His research interests are large-scale transportation system modeling and optimization, and mobility data mining.

Carlee Joe-Wong is an assistant professor of Electrical and Computer Engineering at Carnegie Mellon University. She received her A.B. degree (magna cum laude) in Mathematics, and M.A. and Ph.D. degrees in Applied and Computational Mathematics, from Princeton University in 2011, 2013, and 2016, respectively. Her research interests lie in optimizing various types of networked systems, including applications of machine learning and pricing to cloud computing, mobile/wireless networks, and ridesharing networks. From 2013 to 2014, she was the Director of Advanced Research at DataMi, a startup she co-founded from her research on mobile data pricing. She received the INFORMS ISS Design Science Award in 2014 and the Best Paper Award at IEEE INFOCOM 2012.

Pei Zhang is an associate research professor in the ECE departments at Carnegie Mellon University. He received his bachelor's degree with honors from California Institute of Technology in 2002, and his Ph.D. degree in Electrical Engineering from Princeton University in 2008. While at Princeton University, he developed the ZebraNet system, which is used to track zebras in Kenya. It was the first deployed, wireless, ad- hoc, mobile sensor network. His recent work includes SensorFly (focus on groups of autonomous miniature-helicopter based sensor nodes) and MARS (Muscle Activity Recognition). Beyond research publications, his work has been featured in popular media including CNN, Science Channel, Discovery Channel, CBS News, CNET, Popular Science, BBC Focus, etc. He is also a co-founder of the startup Vibradotech. In addition, he has won several awards including the NSF CAREER award, SenSys Test of Time Award, Google faculty award, and a member of the Department of Defense Computer Science Studies Panel.

Hae Young Noh is an associate professor in the Department of Civil and Environmental Engineering and a courtesy assistant professor in the Department of Electrical and Computer Engineering at Carnegie Mellon University. Her research focuses on indirect sensing and physics-guided data analytics to enable low-cost and non-intrusive monitoring of cyber-physical-human systems. She is particularly interested in developing smart structures and systems to be self-, user-, and surrounding-aware to provide safe and resilient environments and improve users quality of life, while reducing maintenance and operational costs. The result of her work has been deployed in a number of real-world applications from trains, to the Amish community, to eldercare centers, to pig farms. She received her Ph.D. and M.S. degrees in Civil and Environmental Engineering and the second M.S. degree in Electrical Engineering at Stanford University. She earned her B.S. degree in Mechanical and Aerospace Engineering at Cornell University. She received a number of awards, including the Google Faculty Research Awards in 2013 and 2016, the Deans Early Career Fellowship in 2018, and the National Science Foundation CAREER award in 2017.