International Journal of Smart Home
Vol. 10, No. 2, (2016), pp. 219-234
http://dx.doi.org/10.14257/ijsh.2016.10.2.21

Smart Parking by Mobile Crowdsensing

Xiao Chen and Nianzu Liu

School of Mathematics and Information
Shanghai Lixin University of Commerce
chenxiao@lixin.edu.cn,liunianzu@lixin.edu.cn

Abstract

An increasing number of mobile applications aim to realize "smart cities" by utilizing contributions from citizens armed with mobile devices like smartphones. However, there are few generally recognized guidelines for developing and deploying crowdsourcing-based solutions in mobile environments. This paper considers the design of a crowdsensing-based smart parking system as a specific case study in an attempt to explore the basic design principles applicable to an array of similar applications. Through simulations, we show that the strategies behind crowdsensing activities can influence the utility of such applications significantly. Equally important, we show that a certain level of freeriding could be allowed to increase social benefits as long as a reasonable service differentiation mechanism exists. Our findings provide designers with a better understanding of mobile crowdsensing features and help guide successful designs.

***Keywords**: mobile crowdsensing, smart parking, collaborative sensing*

1. Introduction

As the consumer-centric mobile devices like smartphones and GPS navigators become increasingly powerful, versatile and accessible, ordinary people who use such gadgets have the potential to collectively serve as a specialized infrastructure to provide professional data. On one hand, a variety of sensors embedded into smartphones or vehicles allow a person to record observations such as Wi-Fi fingerprints, video clips, road conditions or traffic congestion levels, which can be valuable to others. On the other hand, the use of mobile devices allows people to break time and space barriers to share and exchange information and knowledge. Owning to this fact, the concept of mobile crowdsensing[1] has inspired an array of innovative applications in recent years which include indoor localization [2], searchable video repository construction [23] and privacy protection[24]. In contrast with related terms like participatory sensing[28] and opportunistic sensing[29], mobile crowdsensing has a more general definition so that it might or might not involve human's factors in the data production loop. With the popularity of mobile social networking and the ever increasing computing power of consumer electronics, mobile crowdsensing has been regarded as an effective solution to some tough problems that involve real-time data collection from and coordination among a large number of participants. In particular, mobile crowdsensing can be harnessed to design smart parking systems.

The parking problem has existed in big cities for decades. Studies show that an average of 30% of the traffic in busy areas is caused by vehicles cruising for vacant parking spots [3]. The situation is getting worse in developing countries like China, where the automobile ownership rate has soared recently, while the investment in parking facilities has lagged far behind demand. The additional traffic causes a series of problems such as traffic congestion, air pollution, and energy waste. Some local governments try to mitigate these issues by deploying smart parking systems: systems that employ information and communication technologies to collect and distribute real-time data about parking availability so that drivers can find parking spots quickly.

For example, the city of San Francisco installed thousands of sensors at on-street parking spaces in busy areas to collect and spread parking availability information [18]. Although the benefits of such a centralized approach are immediate, its huge initial investment and maintenance cost inhibits a widespread adoption in most other cities: the average maintenance cost for each sensor monitoring a single parking space is beyond $20 per month [4]. Even in San Francisco, the majority of parking spaces are not covered by the system due to the cost.

This paper studies the properties of crowdsensing in the context of smart parking problem. More specifically, we investigate the use of information crowdsensing for parking guidance like a road navigation system (as a design alternative to lower the cost to install and maintain dedicated infrastructure). It is important to note that Apps like Waze [6] have already demonstrated that crowdsensing through road navigation is feasible and have accumulated millions of users in many countries. Since those apps already collect most of the data we need for parking guidance, our system can be easily implemented as an extension to such existing platforms.

Our proposal improves existing approaches in several ways. First, by integrating crowdsensing and a road navigation system, we make the data collecting tasks much easier for drivers to accomplish. Unlike applications like Open Spot [8], which require drivers to launch them separately for searching parking spots, we only ask drivers for their manual input at the beginning and the end of their trips. By simplifying operations, we are more likely to recruit a larger number of contributors, a factor key to the success of crowdsensing .

Second, since drivers who contribute also benefit from the system, our approach heavily depends on a pattern of mutual assistance, which excludes the complexities caused by monetary rewards [5],[25]. On the one hand, the system is resilient to the existence of free riders as shown in Section 6. On the other hand, because we assume a centralized control over data distribution, the system can create incentives by providing users with different quality of service (e.g., better parking suggestions, request prioritization) based on their contribution records as we will show in Section 4.

Finally, we coordinate the crowdsensing behavior among participants to improve data collection efficiency and system utilization. In contrast to existing approaches that only share information about parking vacancies, our system also tries to identify occupied areas through user's sensor data (or explicit input) so as to help drivers avoid unnecessary cruising. In addition, we assign parking spaces to users dynamically according to their reports to eliminate races between participants. Furthermore, we take a proactive strategy to crowdsource when the current knowledge is limited: more specifically, the system might direct drivers to unexplored areas so that it can expand its knowledge about parking availability in these areas.

Our contributions in this paper are twofold. On the one hand, we demonstrate through simulations that the mobile crowdsensing is a feasible and cost effective approach to deploy a smart parking system. On the other hand, we regard this application as a case study to demystify some rumors that have influenced the design of mobile crowdsensing-based applications for a long time. We find that recruiting more participants may not necessarily lead to a better performance if the crowdsourcer fails to coordinate participants' behavior in the context of these applications. We show that people can provide valuable information by answering very simple questions in a dynamic mobile environment if they are coordinated. We also find that a proper policy to deal with free-riders will improve social benefits without sacrificing the quality of the service.

Copyright ⓒ 2016 SERSC

The rest of the paper is organized as follows: Section 2 positions this work among the related literature; Section 3 describes the parking guidance system and its different strategies to harness crowdsensing; Section 4 provides the details of our incentive mechanism, which is based on contribution evaluation and service differentiation. Sections 5 and 6 present the simulation design and the evaluation results respectively; Section 7 concludes the paper with final remarks.

2. Related Work

The idea of crowdsensing is inherently compatible with humans' daily activities, which have a large amount of participants. An important category of its applications emerge from the field of public transportation. Thanks to the data collected from thousands of drivers, pedestrians and their mobile devices, people are able to choose better traffic routes according to the realtime road conditions on the map, to refill at a gas station with a lower price by GasBuddy [7], or to find a parking vacancy using applications like Open Spot [8]. Similarly, taxi drivers might improve their routes by knowing colleagues' trajectory [9] and commuters can get the real-time transit information from Roadify [10]. One common feature of these scenarios is that the data contributors are meanwhile the consumers of the services.

Although the aforementioned applications have attracted great attention in the market (e.g., as estimated by their download count), they are orthogonal to the research interests of the academic community. As Kanhere discusses [11], current studies in mobile crowdsensing or participatory sensing generally focus more on new applications (e.g., personal health monitoring [12], environmental surveillance [13], or enhanced social media [14]) than on the impact of participating rates and crowdsensing strategies. Issues like privacy preservation, incentive design, or evaluating the trustworthiness of data remain major concerns when deploying these applications into practice.

As far as smart parking is concerned, the majority of existing studies either assume the availability of special devices installed at the parking lots or heavily depend on the accurate data produced by such devices. Systems like [15] and SPARK [16] employ wireless sensors or VANET (Vehicular Ad-hoc Network) devices to collect and disseminate information about parking availability to help drivers find vacant parking spaces. CrowdPark [5] assumes a seller-buyer relationship between drivers, who are going to leave or parking at the lots, to deal with the parking reservation problem. A relevant study [17] tries to realize smart parking by solving an optimal resource allocation problem according to drivers' various parking requirements. Although these solutions could be applicable to large off-road parking garage, they are not suitable for sparsely located on-street parking lots that are not owned by a single entity.

Some remarkable initiatives that realize smart-parking by the infrastructure-based approach include the SFPark [18] project in San Francisco and the CBD in-ground sensors [26,27] in Melbourne. Although the effect is instant, few cities worldwide can afford the high initial investment and the maintenance cost. Alternatively, some naïve apps like Open Spot [8] just mark what users report on the map but failed to recruit enough contributors. We believe there is a viable approach between these two extremes. More specifically, our approach is to introduce a central entity to coordinate participants' behavior in order to make mobile crowdsensing not only an sustainable but also an effective solution to the smart parking problem.

3. System Design

The basic idea behind our design is to build a system that first collects the parking availability information from participant drivers around a certain area and then uses it to guide those drivers, who try to find a vacancy there later. There are several design options for the two steps. A good choice will make a big difference in the usefulness and performance of the final system.

3.1. Assumptions

The goal of smart parking is to inform drivers of a parking vacancy as soon and as close to their destination as possible. To this end, the crowd-based smart parking system collects relevant data from participating drivers, and then uses this data to navigate them to the right parking slots. For convenience, we refer to the drivers who participate in the system as smart parkers (SP) in contrast to those who do not participate as ordinary drivers (OD).

The system consists of three components: central servers, client devices, and smart parkers. Figure 1 shows the relationships between them. We make the following assumptions about how they interact with each others.

Central Servers: The servers collect data from drivers, who report through client devices their destination at the beginning of their trips, their current locations and car speeds while driving and the parking availability on a certain street when they arrive. Using the information collected in real time, the servers maintain a dynamically annotated parking availability map. When a smart parker gets close to his destination, the servers search the dynamic map for potential parking vacancies according to the parker's current location and destination. Then they inform the client device of the search result, which might be either the specific location of the parking spot or the direction of the next turn to the parking spot.

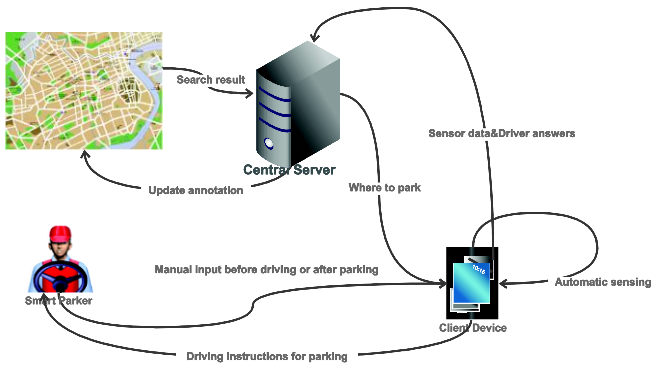

Search result
Central Server
Update annotation
Where to park
Sensor data&Driver answers
Manual input before driving or after parking
Automatic sensing
Smart Parker
Client Device
Driving instructions for parking

Figure 1. Data Flow among the Central Server, the Client Device, and the Smart Parker

In addition to this dynamic data, we also assume that the servers have access to a database, which stores data relevant to parking guidance, such as the parking price, legal periods and locations to park, and statistics about the arrival rate of vehicles and the parking duration around a certain region during a certain period. In fact, an increasing number of cities provide these kinds of data online [20].

Client devices: Drivers have on-board devices that can communicate with the server. They upload geo-tagged data and can download the result of queries regarding parking slots availability. It is reasonable to assume that such devices have GPS capability and Internet connection. A variety of off-the-shelf consumer electronics like smart phones, tablet PCs, and versatile GPS navigators can play the role. The client devices should have a simple user interface that allows smart parkers to input relevant data manually when they are not driving. The devices can also collect geo-tagged sensor data automatically without drivers' intervention when the car is moving. We draw a self-loop on client devices in Figure 1 because the device might process the collected sensor data before sending them to the server.

Smart parkers: Smart parkers are the drivers who have access to the service through their client devices. Like ordinary users of GPS navigators, a smart parker will input her destination before she starts driving. Then she will receive recommendations from the system about potential parking vacancy when she approaches her destination. The smart parker can choose whether or not to follow such recommendations, but the client device will report her cruising trail to the server. At the beginning and the end of a trip the smart parker is expected to answer a question about parking availability in the area by manually operating the client device.

3.2. Problem, Key Questions, and Required Data

Three key questions guide the design of any crowdsensing system: What is the required data? How can we obtain this data through crowdsensing? How can we use the acquired data in the specific application scenario?

In our parking scenario, we model each road segment as a parking lot with several parking spots along it. To realize on-street smart parking, we need to navigate smart parkers to streets that are not fully occupied. In other words, we need to acquire the status of the parking availability along each road segment.

From the server’s perspective, each road segment could have one of the three statuses for its parking availability: available, occupied, or unknown. Initially, the status of all streets is marked as *unknown*. Once information is received the status can switch to *available* or *occupied*.

Unlike smart parkers, ordinary drivers do not provide data when they arrive at or leave from a parking space. In this case the central server cannot notice the change of status in those parking lots. Thus, for all parking spots, we automatically change the status to unknown when a timer expires. The timer length can be derived by statistics or by occupancy prediction [19] and can be adjusted through the observation of the crowdsourced data. In addition to the occupancy status, the system also needs to know the capacity of each on-street parking lot to determine if it can navigate two cars to the same street at the same time.

3.3. Data Acquisition

Suppose a smart parker is about to set off and release a parking spot. She will first inform the agent application of her next destination as well as the maximum walking distance she can accept. When she is approaching the destination, the agent will automatically submit a query to the system. The system will calculate the answer according to the status of parking units. After the agent gets an answer from the server, it will guide the smart parker just like a GPS navigator. If the recommended spot is fully occupied, the agent could resubmit queries several times until an open spot is reached. We assume the agent is able to capture the sensor data like GPS location and velocity. In addition, drivers can also submit manual input via the agent. All uploaded vacancy status is stored as data points on the server side. More details of the involved data structures are explained in Table 1.

Table 1. Related Data Structures

Data Type	Attributes	Meaning
Query	Dest	The destination of the smart parker
	Dist	The maximum acceptable walking distance
	LastLoc	The previously recommended parking location(already occupied) for the same destination

Status of Parking units	UID	ID of the parking unit
	Cap	Observed vacancy capacity
	Exp	Expiration of the status
Data Points	Tim	Timestamp
	Loc	Location information
	Cap	Observed vacancy capacity
	Eve	The code for the event that brings about the data point

Considering the context of smart parking, we propose several crowd-sensing tasks as described in Table 2. A smart parker can fulfill a task by either observing the vacancy status in person or just letting the agent upload predefined values automatically under certain circumstances. Whenever a task is completed, a new data point will be sent to the server. For example, if a smart parker noticed two available spots around where she parked her car, she can report it manually so that a data point like (Tim, Loc, Cap = 2, Eve = A) will be stored on the server side, which is consistent with the task T1. When the car is moving, the agent may determine if the smart parker is cruising for an open slot according to the current speed and the distance from the destination. Given the cruising trails of smart parkers, the server can infer if the parking slots in a certain area are fully occupied.

Table 2. Crowd-sensing Tasks for Smart Parking

Task Name	When	What		How
		Eve	Cap	
T1	Before departure or after arrival	D or A	{0,1} or {0,1,2,3….}	Manual input
T2	Before departure or after arrival	D or A	1	Automatically
T3	Reaching the recommended spot but it is occupied	C	0	Automatically
T4	Moving slowly around the destination	M	0	Automatically

3.4. Parking Guidance Alternatives: Coordinated vs. Uncoordinated

Once the server annotates each street on the map with its parking availability status, the simplest way to do parking guidance is to display the locations of available parking slots on a map directly to all drivers without attempting to coordinate them. However, this uncoordinated approach (also adopted by Open Spot) can lead to several problems.

First, it is usually difficult for drivers to integrate all information on the annotated map to make a good decision when driving. They could always focus on the same parking slots reported by other drivers, which might not be their best choice. Furthermore, when drivers cruise along occupied streets, they cannot help others to avoid such areas which in turn lead to longer cruising time. Due to the uncoordinated nature, smart parkers are less likely to explore unknown areas, where there could be more available parking slots closer to the destination.

To mitigate the problems, we propose to coordinate the drivers (instead of letting them choose where to park by themselves). To eliminate the race between two smart parkers for the same parking spot, we keep track of the capacity of each road segment and navigate smart parkers according to the streets' current available capacity. To find out the parking status around the unknown areas, we assume each unknown street has a capacity of one. Once a street is assigned to a smart parker, its capacity is reduced by one and we only navigate cars to streets with a non-zero capacity. If the assigned street is already fully occupied when the smart parker arrives there, we navigate the car to cruise toward streets with non-zero capacity. This way, we not only help the smart parkers avoid unnecessary cruising but also increase the server's knowledge about unknown streets. The difference between our approach and uncoordinated crowdsensing is shown in Figure 2. In simulation experiments we will show that the coordinated smart parkers have a shorter cruise time and a shorter walking distance than the uncoordinated ones.

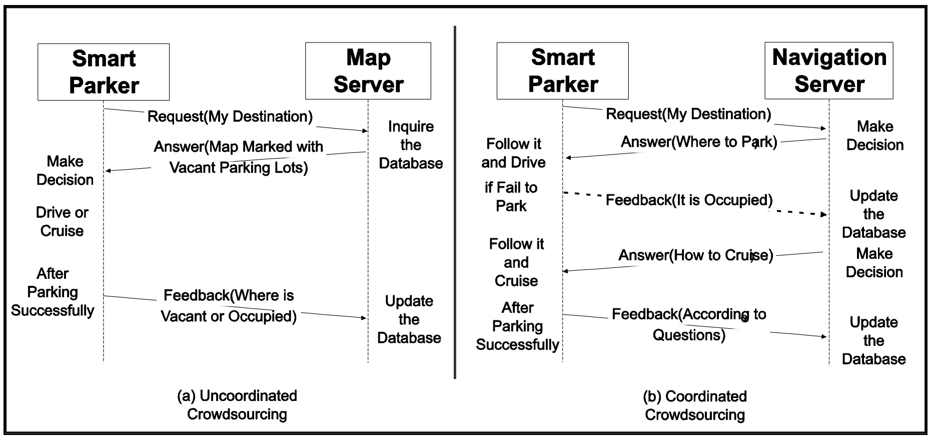


Figure 2. Illustration between Uncoordinated and Coordinated Parking Guidance

4. Incentive Mechanism

A reasonable incentive design is a key factor to develop any sustainable application by crowdsensing. We try to reward participants according to their contribution level so that the incentives will be directly relevant to the application's goal. In other words, the system should first be able to measure the utility or value of a certain data submission and then reward the different contributors accordingly.

4.1 Contribution Assessment

The word *contribution* here refers to the data submission that is relevant to the target application. In our assumed application scenario, a crowd-sensing participant makes a contribution if its data help other smart parkers either find open spots directly or avoid cruising around fully occupied parking areas.

To formalize this basic idea, we need to analyze the relationship between different data points, whose data structure is given in Table 1. Suppose the contributor of the data point *a* is denoted by *SP(a)*. We define the following relationship between two data points:

Definition 1. Let *a* and *b* be two data points submitted by two crowd-sensnig participants, then *b* is said to be attributed to *a*, denoted by *a->b*, if the following conditions are satisfied (notations are presented in Table 1):

a) a.Cap>0

b) b.Eve=A

c) 0 <b.Tim-a.Tim<t(a.Tim,a.Loc,a.Cap)

d) Distance(b.Loc, a.Loc)<ObservableRadius

The first two conditions indicate that *SP(a)* found some vacancies where *SP(b)* would park her car later. Since *a* and *b* are temporally and spatially related according to the conditions c) and d), we can determine that *a* is a useful data point for guiding *SP(b)*. The function *t(Tim,Loc,Cap)* calculates how long the vacancy status reported in the data point *a* should remain valid in a certain temporal-spatial situation. By condition d), we mean that the vacancy status at location *b.Loc* can be observed from location *a.Loc*.

A smart parker's contribution is measured by the ratio between how many times she helps others and how many times she asks for help during a certain period. Suppose each time the server receives a query, it generates a new data point whose value for *Cap* is -1. Then we have the formal definition for the contribution measurement as follows:

Definition 2. Let *D* be the set of all data points, the contribution ratio of smart parker s during the period T is defined as:

$$CR_T(s) = \frac{|\{b \mid a -> b, a \in D, b \in D, a.Tim \in T,\ SP(a) = s\} \cup \{a \mid a.Eve = C, a.Tim \in T, a \in D, SP(a) = s\}|}{|\{a \mid a.Cap = -1, a \in D, a.Tim \in T, SP(a) = s\}|}$$

It is possible that multiple data points are attributed to the same one so that this value could be greater than one for some experienced smart parkers. However, a "smart" parker can also report false vacancies, which do not exist, without lowering her CR value. For this reason, we make the following definitions to record smart parkers' reputation.

Definition 3. Let *a* and *b* be two data points from the set *D*, then *b* is said to be inconsistent with *a*, denoted by *a*-<*b*, if the following conditions are satisfied:

a) We cannot find a data point c∈D such that a->c.
b) a.Cap>0
c) b.Eve=C
d) 0 <b.Tim-a.Tim<t(a.Tim,a.Loc,a.Cap)
e) b.Loc = a.Loc

In definition 3, c) means that *SP(b)* cruised around the location *b.Loc* but failed to find any vacancy. The values *CR* and *RP* for each participant should be updated regularly so that they are motivated to keep contributing useful data. Our system will determine how to respond to a smart parker's query according to not only her contribution ratio but also her reputation, which is calculated as follows.

Definition 4. Let *D* be the set of all data points, the reputation of smart parker *s* during the period *T* is defined as:

$$RP_T(s) = 1 - \frac{|\{b \mid a -< b, a \in D, b \in D, a.Tim \in T,\ SP(a) = s\}|}{|\{a \mid a.Cap = -1, a \in D, a.Tim \in T, SP(a) = s\}|}$$

Suppose table 3 is a list of some simplified sample data points submitted to the server. The first column represents the usernames of the contributors while the others correspond to the four fields of each data point respectively. The data points are grouped by the location and then listed in a chronological order. The first row means that Alice reported 3 open spots around the east side of the theater before leaving at 10am. By the second row, we know that Bob successfully parked his car 5 minutes later at the same place and reported two vacancies. If all three vacancies there at that time can be hardly occupied within 5 minutes, we can say that the second data point is attributed to the first one by the definition 1. In other words, we believe Alice helped Bob find his slot. However, we don't think Carol's data point is related to the first two ones because we can hardly predict if these vacancies still remain available after 4 hours if it's a popular place.

Table 3. Fictitious Data Point List

User Name	Location	Time	Capacity	Event
Alice	East side of the Theater	10am	3	D
Bob	East side of the Theater	10:05am	2	A
Carol	East side of the Theater	14:00pm	1	A
Denny	West side of the Library	17:00pm	6	A
Ella	West side of the Library	17:10pm	0	C

By the last two rows, we know Ella's data point is inconsistent with Denny's because the former cannot find any vacancy as reported by the latter 10 minutes before. This would have a negative influence on Denny's reputation as defined in definition 3. Since there are all kinds of uncertainties in the real world, the value of a person's reputation is reliable only if it's assessed in a long term.

4.2. Service Differentiation

Since not all drivers participate in the crowd-sensing, the system does not have a guaranteed data source as in most centralized systems. For example, an ordinary driver could occupy a vacancy silently without informing the system of the status change. To deal with this issue, we group adjacent parking slots into parking units whose vacancy status changes less frequently than a single slot. We also set an expiration time for the status of each parking unit on the server side to avoid the influence of outdated status. Furthermore, the server can proactively assign tasks to smart parkers during their navigation process in order to update and expand its knowledge base. This strategy forms the basics of the service differentiation.

When receiving a query, the server will respond with the location of a recommended parking unit. If the status of the unit expired or is unknown to the system, the recommendation then becomes a task assigned to the smart parker, who at this moment will act as a data producer. The data reported from the specified location will help the system enhance its knowledge base. In return, the system regards each completed task as a contribution of the involved participant. We show the basic pseudo code of the query processing in Figure 3.

To serve smart parkers in a reasonable manner, the system should follow two principles when determining whom to assign a task and what task to assign:

- A smart parker with a high contribution ratio and a good reputation record should have the priority to be served directly.
- The assigned tasks should point to a better parking position which deserves the effort for checking.

```
Process(Query q) {
  DataPoint dp=newDataPoint();
  dp.Tim=CurrentTime
  If q.LastLoc=Null then
    dp.Loc=q.Loc; dp.Cap=-1
  Else {
    dp.Loc=q.LastLoc; dp.Eve=C; dp.Cap=0
    Location l1=getValidVacancy(q.Loc)
    Location l2=getTaskLocation(q.Loc)
    If Dist(l1,q.Loc)<Dist(l2,q.Loc) then l2=l1
      If RP(SP(q))*CR(SP(q)) >AvgValue AND
        Distance(l1,q.Loc)<q.Dist then
        AnswerWith(l1)
      ElseIf Distance(l2,q.Loc)<q.Dist then
        AnswerWith(l2)
    Else
      AnswerWith(NoSuitable)
  }
}
```

Figure 3. Pseudo Code for Processing Queries

5. Simulation Methodology

We test the feasibility of our system by simulating a real-world street area, where vehicles arrive frequently. By simulations we can discover the best crowdsensing strategy in order to improve the user experience and the system performance. This section describes the simulation settings in detail.

5.1. Simulation Environment

To simulate the crowdsensing-based system in the context of smart parking scenario, we need to take care of two aspects. On one hand, the simulations should reflect features in realistic road traffic environment like road layout, car following patterns, and individual driving behaviors. On the other hand, the simulation environment should be configurable to apply the different design options discussed in section 3. Since no existing simulation environments can satisfy all requirements, we modified an open source road traffic simulator, SUMO [21], to meet our needs.

SUMO is a microscopic road traffic simulator, which allows simulating thousands of vehicles moving through a road network. The simulator is capable of capturing the geospatial properties of each vehicle in motion like location and speed at any moment. This corresponds to our assumption about smart parkers: they should be able to report such information to the server when driving. However, the existing environment heavily depends on predefined configuration files to determine the departure time and the route of each vehicle.

To simulate the dynamic scenario, in which vehicles arrive according to a Poisson process and cruise around for an open spot to park, we integrate the logic of vehicle generation and routing into the simulator. In addition, the adapted simulator adds the parking capacity as a new property to each street in the road network so that smart parkers will keep cruising in search of open parking slots until they enter a street with a non-zero parking capacity. Furthermore, we have implemented the data collection process and parking navigation inside SUMO to reflect the different crowdsensing strategies mentioned in section 3.

5.2. Simulated Scenario and Parameter Setting

Scenario: The simulations aim to evaluate the feasibility of the aforementioned crowdsensing system in a simple but realistic scenario, where hundreds of vehicles are heading for the same destination during a short period of time and few cars leave the parking lots at that time. This often happens around office buildings and park-and-ride facilities [22] during rush hours or at a stadium before a game kicks off. This scenario helps us focus on the impact of different design choices for the crowdsensing system rather than on the statistics related to parking lot usage around a certain area.

Parameter setting: The road network in our simulations is modeled as a 1 km^2 region divided into a 9x9 grid by four-lane bidirectional streets. Each road segment has a parking capacity of 5 for either side of the street. In the simulator, the block in the center is assumed as a common destination and everyone tries to park close to it in order to reduce the walking distance. In each round of simulation, a sequence of about 1,000 vehicles enters the map according to a Poisson process. The average arrival rate is set to one car every 15 seconds.

The simulator determines whether a new coming driver is a smart parker by a certain probability so that it is possible to control the approximate ratio between the two groups of drivers. If an ordinary driver cannot find an open spot on the destination street, he will have to cruise around randomly until he can find one somewhere else. The speed limit for normal driving is 50km/h while the cruising speed is about 10km/h.

When a smart parker moves close to the desired destination, the server will show her suggestions about the available parking place. If she follows the server's suggestion but reaches a fully occupied on-street lot, she also needs to cruise. However, the parking guidance will help during the cruising if we adopt a coordinated guidance strategy. We run each simulation from 5 to 35 times and plot the average value.

6. Evaluation Results

There are two criteria to evaluate the performance of a smart parking system: the average walking distance from the located parking spot to the actual destination (measured in 'blocks' – i.e., the distance between two crossroads) and the average cruising time to find a parking spot given a certain number of arriving cars and available parking spots. We record these figures in our simulation experiments to compare the effect of different crowdsensing strategies.

In the following simulations, we explore the impact of different crowdsensing policies from three aspects: the impact of coordination among participants; the impact of data accuracy; and the social impact of freeriding.

6.1. Uncoordinated VS Coordinated Crowdsensing

The first question we focus on is: Do smart parkers outperform ordinary drivers regardless of the crowdsensing strategy used by the system? We first assume that the system adopts a pure uncoordinated strategy so that each smart parker just follows a predecessor who located the closest vacancy to the destination and reported open spots around it.

Figure 4(a) compares two groups of drivers with regard to the average walking distance. We collect the data as the participation rate (i.e., the ratio of smart parkers in the system) increases from 10% to 50% of the driver population. As Figure 4(a) shows, uncoordinated crowdsensing leads to longer walking distance for smart parkers than for ordinary drivers. Since the system does not provide smart parkers with a global view around the region, they tend to miss potential vacancies closer to their destination.

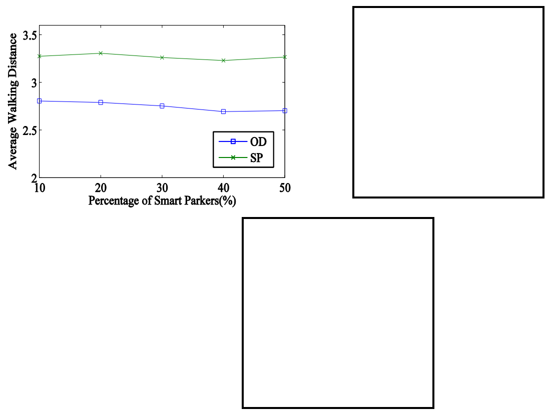


Figure 4. Performance Comparison between Ordinary Drivers and Smart Parkers Adopting an Uncoordinated Crowdsensing Approach. The Walking Distance in (a) and (b) is Measured in Number of Blocks Away from the Central Destination. Error Bars Here Indicate the 95% Confidence Interval

Smart parkers might not always follow directions to the recommended spots, which are far away from the destination. Thus we next assume that they choose to cruise by themselves and ignore the system's recommendations if they are more than three blocks away from the destination, which is the median value for the walking distance here. The resulting walking distance and average cruising time are shown in figure 4(b) and 4(c) respectively. Although smart parkers don't lose to ordinary drivers in terms of average walking distance this time, about 40% of them spend more search time than ordinary drivers.

The previous figures show that the uncoordinated approach, also used by Google's Open Spot, fails to help users do a better job than ordinary drivers in the search of parking spots no matter how many drivers participate.

We assume now a coordinated approach where the system collects information by assigning tasks in table 2. In addition, it assigns smart parkers to explore unknown areas and helps them cruise more efficiently by avoiding occupied streets. As the figure 5(a) shows, such an approach achieves a lower average walking distance for smart parkers.

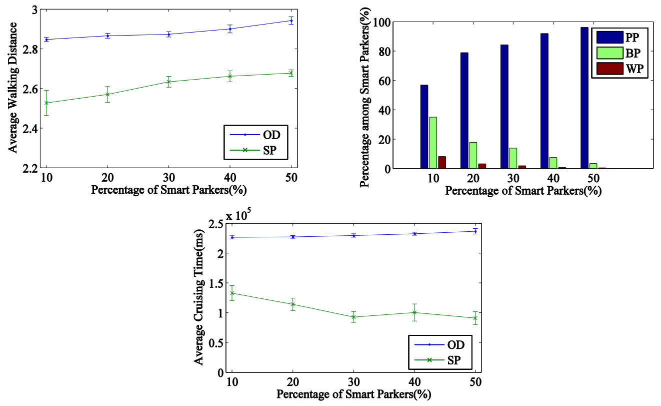


Figure 5. Performance of Smart Parkers when their Behavior is Coordinated in the Crowdsensing System. Error Bars here Indicate the 95% Confidence Interval

As far as the cruising time is concerned, we divide the smart parkers into three groups: the majority of them can find an open spot immediately according to the system's navigation and we call them perfect parkers (PP). Those who still need to cruise but spend less time than the average cruising time of ordinary drivers are referred to as better parkers (BP). The rest are called worse parkers (WP) as they spend more time cruising. Figure 5(b) shows the change of the composition of smart parkers as more drivers participate. More than 90% of the smart parkers do not need to cruise when the membership covers about 40% of all drivers. For BP and WP, we calculate their average cruising time and plot the result in figure 5(c). All these figures show that the coordinated crowdsensing is more effective in the smart parking scenario.

6.2. Impact of Various Design Options Leading to Increased Usability

The second question we try to study concerns both developers and users: Could the data collection process be simpler (user friendly) while still achieving the application goals? To make it clear, we let smart parkers answer different questions to report relevant information to the server and then estimate their impact on system efficiency (we use the number of perfect parkers for evaluation). We plot the results in figure 6(a) with each line for a specific question. The question Q1 asks the drivers for the specific number of vacancies around when they arrive at or depart from their spots. In contrast, Q2 stands for a Yes-No question and Q3 assumes a default answer of 1 to all questions. The figure reflects that the answers to a Yes-No parking availability question provide sufficient information for the server to implement a useful navigation service. Next, we repeat the experiments without inferring the occupancy through cruising vehicles. By comparing figure 6(a) and 6(b), we find that the information inferred through cruising cars(task T3 and T4 in Table 2) is helpful when only a few smart parkers participate but its importance diminishes as more drivers join the system.

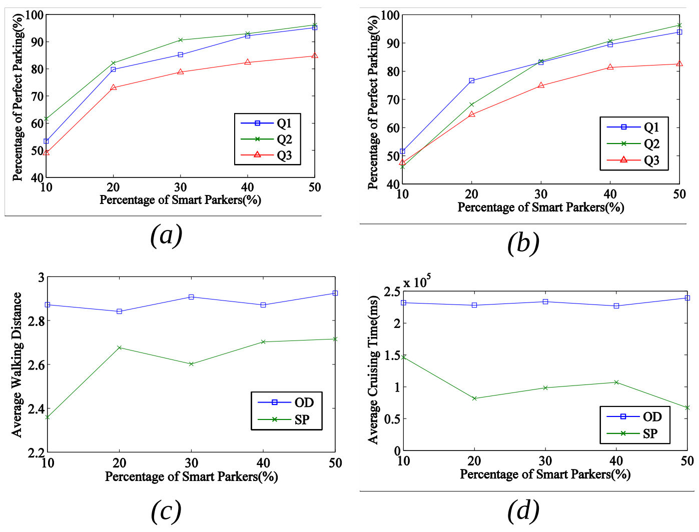


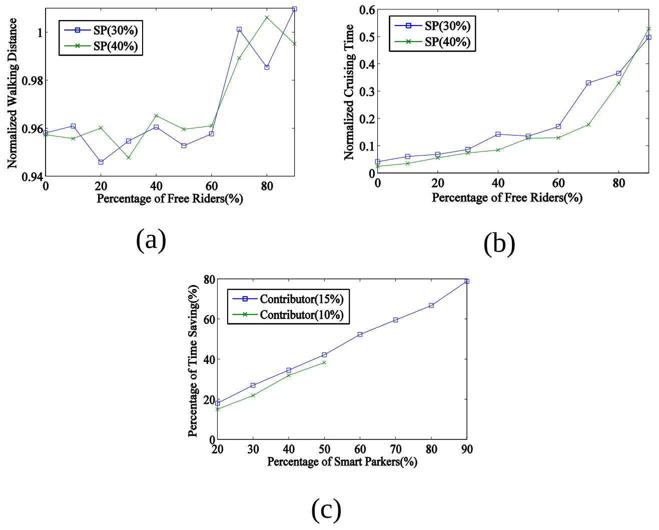
Figure 6. Influence of Various Crowdsensing Options to the Performance of Smart Parking System. In (a) and (b), We Assign Different Tasks after Smart Parkers Find their Spots. We First Turn on in (a) and then Turn Off in (b) the Option to Infer Occupancy through Cruising Cars. In (c) and (d), we Only Ask Drivers if or Not there are Additional Spots on the Street Where They Parked the Cars

To increase confidence in the preliminary conclusion, we assume that smart parkers only take task T2 and no information is inferred when they are cruising. In other words, the server only asks the Yes-No question this time and makes no inference about occupied streets when smart parkers are cruising. We compare smart parkers with ordinary drivers again with regard to the average walking distance and cruising time in figure 6(c) and 6(d) respectively. Since the majority of smart parkers (at least 50% of them) do not cruise at all, we only plot in figure 6(d) the average cruising time for those who need to cruise, namely the better parkers(BP) and the worse parkers(WP). The figures show that the crowdsensing system is still effective even when participants only answer simple questions.

6.3. The Impact of Free Riders

The last set of simulations deals with another realistic question: How should the system handle free riders? In the context of our system, free riders are those drivers who only want to take advantage of the service but refuse to answer any question. As a part of the feasibility evaluation, we need to evaluate how tolerant the crowdsensing system is to freeriding and decide how to handle them: to tolerate them or just to exclude them.

The average walking distance among all drivers and the average cruising time of ordinary drivers do not change much across all experiments. Therefore we use them as a benchmark to test if smart parkers can still find the open spot quickly and park closer to their destination even in the presence of free riders. In the following simulations, we only ask Yes-No question without data inference during drivers' cruising. We assume that 30% to 40% of all drivers are smart parkers. Among the smart parkers, the percentage of free riders grows from 10% to 90%. We plot the normalized walking distance for smart parkers in figure 7(a), namely the average walking distance of the smart parkers divided by the average walking distance among all drivers. Similarly, we plot the normalized cruising time for smart parkers in figure 7(b). The figure shows that the navigation system works well until the percentage of free riders exceeds 60%. In other words, we can infer that the quality of the service is still acceptable as long as at least 12% of all drivers are willing to contribute.



Fgure 7. Tolerance for Free Riders in the Crowdsensing-based Smart Parking System. In (a) and (b), we assume two fixed percentages of smart parkers among all drivers and record their performance as more smart parkers become free riders. In (c), we assume two fixed percentages of contributors among all drivers and record how much cruising time can be saved as more people become free riders.

As mentioned, the average cruising time of ordinary drivers does not change much as the share of smart parkers grows. This means that the overall cruising time of all drivers will remain a constant if the smart parking system is not available. If we use this cruising time as a measure of social welfare, allowing freeriding could boost social welfare, which means the overall time and fuel consumption will be reduced. In the following simulations, we assume the contributors account for 10% or 15% among all drivers while the percentage of smart parkers (including both contributors and free-riders) grows from 20% to 90% of the total population. Then we calculate the percentage of the time saving as long as the system is able to keep smart parkers closer to their destination. As figure 7(c) shows, when contributors and free riders account for 15% and 35% of the population respectively, above 40% of the overall cruising time can be saved. As we mentioned before, it is difficult to maintain the quality of the service if the number of free-riders keeps growing while only 10% of all drivers contributing to the system. However, if the amount of contributors reaches 15%, the system can accommodate much more free-riders and reduce the overall cruising time significantly. These results show that we can allow free riders to exist if there are enough contributors in the crowdsensing system.

7. Conclusion

Mobile crowdsensing is an increasingly popular mechanism to realize applications that harness a large volume of real-time data to improve daily life efficiency. Crowdsensing, however, brings several new issues that arise only in the context of participatory, peer-to-peer systems. In this paper, we take smart parking as a use-case and explore the possible design options to deal with these issues. In particular, our study leads to the following findings:

First, crowdsensing in a mobile environment can lead to ‘herd’ behavior rather than collective intelligence since each participant only has a limited view of his surroundings and a global picture of the physical world is not realizable. To deal with this issue, we propose ‘coordinated crowdsensing’, in which a server integrates all information from participants and encourages them to explore unknown area. Our simulations show that the coordinated crowdsensing is an effective approach in a mobile environment.

Second, the participation rate is more important than the volume of information each individual contributes. Our simulations show that, when the membership rate of a crowdsensing system passes a certain threshold, the outcomes remain stable even if each individual contributes only a limited amount of information. However, if the participation rate is low, a sophisticated data collection mechanism becomes necessary to compensate the lack of data sources.

Last, the crowdsensing-based application might continue to increase social welfare by tolerating free riders, as long as it can maintain a moderate level of contribution among participants. In the context of mobile crowdsensing, free riders could reduce the quality of the crowdsensing-based service as they might change the status of the physical environment without reporting new information. However, the aggregated social benefit for all participants could still rise significantly (at the cost of a slightly degraded service quality) as long as a certain percentage of the members keep contributing their data.

References

[1] R. K. Ganti, F. Ye, and H. Lei. "Mobile crowdsensing: current state and future challenges." Communications Magazine, **(2011)**, pp. 32 - 39.

[2] S. Yang, et al., "FreeLoc: Calibration-Free Crowdsourced Indoor Localization", INFOCOM, Proceedings IEEE , **(2013)**.

[3] P. White, "No Vacancy: Park Slopes Parking Problem And How to Fix It," Http://www.Transalt.org/newsroom/releases/126, .

[4] Sarah Kessler, "How Smarter Parking Technology Will Reduce Traffic Congestion," Http://mashable.com/2011/04/13/smart-Parking-Tech/, **(2011)**.

[5] Tingxin Yan, "CrowdPark:A Crowdsourcing-based Parking Reservation System for Mobile Phones." UMASS Technical Report., Tech. Rep. UM-CS-2011-001, **(2011)**.

[6] Waze, Http://www.Waze.Com/, **(2014)**.

[7] GasBuddy, "Find Low Gas Prices in the USA and Canada", Http://gasbuddy.Com, .

[8] J. Kincaid, "Googles Open Spot Makes Parking A Breeze, Assuming Everyone Turns Into A Good Samaritan." Http://techcrunch.com/2010/07/09/google-Parking-Open-Spot/, .

[9] Bin Li, "Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset," Pervasive Computing and Communications Workshops, **(2011)**.

[10] N. Lamba, "Social Media Trackles Traffic," http://www.Wired.com/autopia/2010/12/ibm-Thoughts-on-a-Smarter-Planet-8/, **(2010)**.

[11] S.S. Kanhere, "Participatory Sensing: Crowdsourcing Data from Mobile Smartphones in Urban Spaces," in Mobile Data Management (MDM), **(2011)**.

[12] S. Reddy, et al., "Image Browsing, Processing and Clustering for Participatory Sensing: Lessons from a DietSense Prototype", Workshop on Embedded Networked Sensors, **(2007)**.

[13] M. Mun, "PEIR, the Personal Environmental Impact Report, as a Platform for Participatory Sensing Systems Research," MobiSys’09.

[14] E. Miluzzo, et al., "Sensing Meets Mobile Social Networks: The Design, Implementation and Evaluation of the CenceMe Application," ACM SenSys, USA, **(2008)** .

[15] Jatuporn Chinrungrueng, "Smart Parking: An Application of Optical Wireless Sensor Network," in Applications and the Internet Workshops, **(2007)**.

[16] Rongxing Lu, "SPARK: A New VANET-Based Smart Parking Scheme for Large Parking Lots," in INFOCOM 2009, IEEE, **(2009)** , pp. 1413-1421.

[17] Y. Geng and C.G. Cassandras, "A new “smart parking” system based on optimal resource allocation and reservations," in Intelligent Transportation Systems (ITSC), **(2011)**.

[18] SFPark, “San francisco parking sensor locations,” http://sfpark.org/howit-works/the-sensors/, 7, **(2012)**.

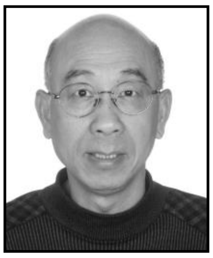
[19] M. Caliskan, "Predicting Parking Lot Occupancy in Vehicular Ad Hoc Networks," in Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th, **(2007)**, pp. 277-281.

[20] Anonymous "Parking Meter Rates and Time Limits," Http://vancouver.ca/vanmap/p/parking Meter.Htm, .

[21] M. Behrisch, "SUMO - Simulation of Urban MObility: An Overview", SIMUL 2011, The Third International Conference on Advances in System Simulation, **(2011)**.
[22] F. W. K. Tsang, "Improved modeling of park-and-ride transfer time: Capturing the within-day dynamics," Journal of Advanced Transportation, vol. 39, **(2005)**.
[23] P. Simoens, "Scalable crowd-sourcing of video from mobile devices." Proceeding of the 11th annual international conference on Mobile systems, applications, and services ACM, **(2013)**.
[24] Y. Agarwal, and M. Hall. "ProtectMyPrivacy: Detecting and Mitigating Privacy Leaks on iOS Devices Using Crowdsourcing", Proceeding of Annual International Conference on Mobile Systems Applications & Services, **(2013)**, pp. 97-110.
[25] Y. Sun,, "Sensing processes participation game of smartphones in participatory sensing systems." Sensing, Communication, and Networking (SECON), 2014 Eleventh Annual IEEE International Conference on IEEE, **(2014)**, pp. 239-247.
[26] C. of Melbourne, “Melbourne cbd in-ground sensor implementation map,” http://www.melbourne.vic.gov.au/ParkingTransportandRoads/Parking/Pages/InGroundSensors.aspx, 5, **(2012)**.
[27] ——, “City of melbourne,” http://www.melbourne.vic.gov.au/, **(2014)**.
[28] J. Burke, “Participatory Sensing,” Wksp. World-Sensor-Web, Collocated with ACM SenSys, **(2006)**; http://www.sensorplanet.org/wsw2006/.
[29] N. Lane, “A Survey of Mobile Phone Sensing,” IEEE Commun. Mag., vol. 48, no. 9, **(2010)**, pp. 140–50.
[30] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges." Communications Magazine IEEE 49, vol. 11, **(2011)**, pp. 32 - 39.

Authors

Xiao Chen, He received his B.S. and Ph.D. degree in Computer Science and Engineering from Shanghai Jiao Tong University, Shanghai, China, in 2002 and 2008 respectively. He is a Lecturer at Shanghai Lixin University of Commerce. His main research interests include Intelligent Traffic System, Mobile Crowdsourcing Applications, and Architecture of the Internet.



Nianzu Liu, He is a professor in the School of Mathematics and Information Science, Shanghai Lixin University of Commerce, Shanghai China. His research interests include Computational Graph Theory, Database Application, and Algorithm and Data Structure. He has more than 40 publications, which include textbooks and research papers in related fields. He is also the chief investigator of several national funded projects.

Copyright ⓒ 2016 SERSC