Crowdsourcing for On-street Smart Parking

Xiao Chen, Elizeu Santos-Neto, Matei Ripeanu
Department of Electrical and Computer Engineering, University of British Columbia
{xiaoc, elizeus,matei}@ece.ubc.ca

ABSTRACT

Crowdsourcing has inspired a variety of novel mobile applications. However, identifying common practices across different applications is still challenging. In this paper, we use smart parking as a case study to investigate features of crowdsourcing that may apply to other mobile applications. Based on this we derive principles for efficiently harnessing crowdsourcing. We draw three key guidelines: First, we suggest that that the organizer can play an important role in coordinating participants', a key factor to successful crowdsourcing experience. Second, we suggest that the expected participation rate is a key factor when designing the crowdsourcing system: a system with a lower expected participation rate will place a higher burden in individual participants (e.g., through more complex interfaces that aim to improve the accuracy of the collected data). Finally, we suggest that not only above certain threshold of contributors, a crowdsourcing-based system is resilient to freeriding but, surprisingly, that including freeriders (i.e., actors that do not participate in system effort but share its benefits in terms of coordination) benefits the entire system.

Categories and Subject Descriptors

C.2.4 [Computer Systems Organization]: COMPUTER-COMMUNICATION NETWORKS - Distributed Systems - Distributed applications

General Terms

Algorithms, Design, Economics, Human Factors

Keywords

Mobile crowdsourcing, smart parking, collaborative sensing.

1. INTRODUCTION

In many application contexts, crowdsourcing has reintroduced humans into the information processing loop. Several success stories [1,2] show that when ordinary people's knowledge, experience, or creativity are aggregated, they have the potential to replace the roles of the most powerful computers, the most knowledgeable experts, or the most talented designers. Currently, the majority of crowdsourcing-based applications focus on harnessing collective intelligence via web applications.

However, as the wireless communication infrastructure and mobile applications keep thriving these years, the influence of crowdsourcing can have direct impact on applications affect our physical world as well. When equipped with a smartphone and constantly connected to the wireless network, everyone is able to collect real-time data about the physical world either through her observation and manual input or by the sensors in the device. Therefore, mobile crowdsourcing enables data collection through thousands or millions of such intelligent probes and collects data primarily from the surroundings of people's everyday life. Such collaborative data collection enables the design and implementation of services that are helpful to each individual in our increasingly connected society. One example is the realization of smart parking.

The parking problem has existed in big cities around the world for decades. Studies show that an average of 30% of the traffic [3] in busy areas is caused by vehicles cruising for vacant parking spots. The situation is getting even worse in developing countries like China, where the number of private cars soars without sufficient parking facilities. The extra traffic causes a series of problems such as traffic congestions and accidents, air pollution and waste of fuel. Some local governments try to mitigate these issues by means of smart parking. In a nutshell, they try to employ information and communication technologies to collect and distribute the real-time data about parking availability so that drivers can spot the right parking place quickly. For example, the city of San Francisco installed thousands of sensors to on-street parking spaces in its busy areas for parking management. While the effect of such a centralized approach is immediate, its huge initial investment and maintenance cost inhibits a widespread adoption in most other cities.

In this paper, we derive design guidelines for a mobile crowdsourcing framework that integrates the functionality of parking guidance into a road navigation system. To be able to generalize our finding to other crowd-sensing scenarios, we assume that there is no additional sensing infrastructure deployed to monitor parking spaces. The only sensing device our solution requires is the road navigation system each driver alreay uses. While we are not the first to propose crowdsourcing for smart parking, our proposal has several advantages over existing approaches. First, by crowdsourcing through a road navigation system, we eliminate unnecessary manual operation during driving, which complies with safety regulation in most countries. Unlike applications such as Open Spot [8], which requires drivers to launch it 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 key success factor for crowdsourcing.

Second, since drivers who contribute can also benefit from the system, our approach heavily depends on a pattern of mutual assistance, which excludes the complexities caused by monetary reward [5]. On the one hand, the system has a high resilience to the existence of free riders as shown in the following sections. On the other hand, because we assume a centralized control over the data distribution, we can differentiate users with different quality of service based on their contribution records. It could serve as a supplement mechanism to enforce fairness and motivate participants to contribute.

Third, we coordinate the crowdsourcing behavior among participants to boost the efficiency of the data collection and 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 different users dynamically according to their reported capacity to eliminate the race between participants. Furthermore, we take a proactive strategy to data collection when the current knowledge is limited. More specifically, we might sugget that drivers navigate to unexplored areas so that we can expand the system's knowledge about parking availability to cover more areas.

Our contributions in this paper are twofold. On one hand, we propose a feasible and economical approach that applies mobile crowdsourcing to realize smart parking. On the other hand, we view smart parking as a case study for the more general problem of crowd-sensing: gathering information about a large area from a set of actors armed with mobile sensing devices. For example, we study the relationship between the performance of the crowdsourcing system and its membership, the influence of the freeriding, and the difference between coordinated and uncoordinated crowdsourcing.

The rest of the paper is organized as follows. Section 2 introduces some related works about mobile crowdsourcing and its application in the field of transportation management. In section 3, we describe the parking guidance system and its different strategy for doing crowdsourcing in details. The simulation and the result data is explained and analyzed in section 4 and 5 respectively. We conclude the paper and discuss the future works in section 6.

2. RELATED WORK

An drive behind mobile crowdsourcing applications is the huge demand for a transportation related services, that make everyday life convenient. For example, thanks to data crowdsourced through thousands of mobile devices, drivers are able to pick a better route to avoid a road segment congested five minutes earlier (using by Waze [5]), refill at a gas station with a lower price (using GasBuddy [7]), or find a parking spot (using OpenSpot [8]). Similarly, taxi drivers improve their driving strategy by knowing their colleagues' trajectory [9]; commuters can get the real-time transit information (using Roadify [10]) and feel less anxiety when waiting for the bus. One commonality shared by these scenarios of mobile crowdsourcing is that most consumers of the services also have the ability to contribute data. Therefore, these crowdsourcing-based services can become self-sustainable if they can attract a enough users.

The aforementioned applications have gained high popularity and widespread deployment. At the same time the academic community focuses on [11] novel applications like personal health monitoring [12], environmental surveillance [13] or enhanced social media [14]. Since people are not naturally motivated to contribute their data in these applications, issues like privacy preserving, incentive design or evaluating trustworthiness of data become their major concerns. However, few studies aim to reveal what make existing mobile crowdsourcing applications more successful than others.

As far as smart parking is concerned, the majority of the related work either depends on smart gadgets connected by a communication infrastructure installed at the parking lots or require all drivers to comply with the same protocol when they are going to park. Systems like [15] and SPARK[16] employ wireless sensors and VANET devices respectively to collect and disseminate information about parking availability in order 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 drivers' various parking requirements. However, the reservation-based solutions might complicate drivers' operation and could collapse if only a few drivers participate.

One remarkable initiative that realizes smart-parking by the infrastructure-based approach is the SFPark [18] project in San Francisco. Although the benefit is obvious, few cities in the world can afford the high initial investment and maintenance cost. On the other hand, some pure crowdsourcing-based solutions 5e.g., OpenSpot [8]) are emerging but fail to solve the problem effectively today. We believe there should be a viable approach between these two extremes.

3. SYSTEM DESIGN

3.1 Assumptions

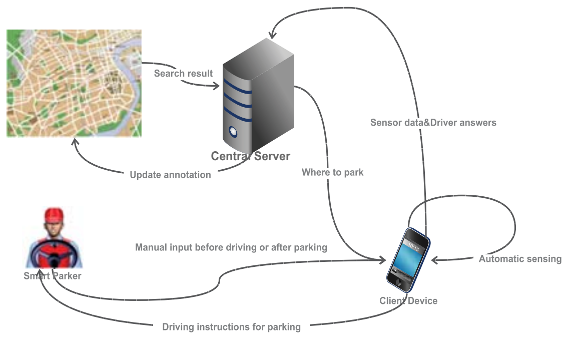
The ultimate goal of smart parking is to help drivers find the parking vacancy as soon and as close to their destination as possible in order to save the time and fuel spent cruising, to reduce unnecessary walking, and to improve the traffic situation and the environment. To this end, the crowd-based smart parking system collects relevant data from drivers, who participate in the crowdsourcing, and then use these data to navigate them to the right parking slots. For convenience, we refer to these drivers hereinafter as *smart parkers* (SP) in contrast to ordinary drivers. The whole system consists of three entities: a server, client devices, and smart parkers. We make the following assumptions about their responsibility or functionality.

Central Server: The server should be operated by a single party like a company or institution in a centralized way so that there is no direct data exchange between two smart parkers. The server collects crowdsourced data, such as driver's current location and destination, car speed, and parking availability on a certain street, through client devices in real time in order to maintain a dynamically annotated street map. When a smart parker drives close to his destination, the server searches the dynamic map for potential parking vacancies according to the parker's current location and destination. Then the server informs the client device of the search result, which might be either the specific location of the parking spot or the direction of the next street to drive to.

In addition to the dynamic data, we also assume the server has access to the static data that are relevant to parking guidance. Such data include the parking price, legal periods and areas to park, and statistics about the arrival rate of vehicles and parking rate around a certain region. They not only help the server deal with special parking requirements but also serve as important parameters for predicting parking lot occupancy [19]. An increasing number of cities are providing this data online [20].

Client devices: Client devices are on-board devices that can communicate with the server. They upload geo-tagged data and then download the search result, which is converted to graphical or voice instructions to navigate the smart parkers to the right parking spot. Thus they should have GPS capability and an access to the Internet through direct or indirect cellular connectivity. A variety of off-the-shelf consumer electronics like smart phones, tablet PC and some versatile GPS navigators can play this role. The client devices have a simple user interface for 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.

Smart parkers: Smart parkers are drivers who have access to the service through their client devices. Like ordinary users of other GPS navigators, a smart parker needs to input her destination before she starts driving. When she approaches the target place, she can find the recommended direction to a potential parking slot. The smart parker can choose whether or not to follow such instructions but the client device will report his cruising trail to the server. After the car is completely parked or before the car starts, the smart parker is expected to answer a question through manual input on the client device, which is about the parking availability around her according to her observation.



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. Insert caption to place caption below figure.

Figure 1 shows the relationships between the three entities. The arrows represent the data flows between them. We draw a self-loop on client devices because the device might process the collected sensor data before sending them to the server.

3.2 Basic Questions and Required Data

The design of any crowdsourcing system is deeply related to the answer to three questions: what is the required data, how to obtain it through crowdsourcing, and how to use it in the specific application scenario. Furthermore, the features of the required data usually determine the complexity of the whole system.

In the on-street smart parking scenario, we view 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 (lots) that are not fully occupied. To this end, we need to acquire the parking availability status 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* and *Unknown*. Initially, the status of all segments sis marked as *Unknown*. According to the crowdsourced data from smart parkers, the status could switch to or between *Available* and *Occupied*. Since not all drivers are smart parkers and ordinary drivers don't notify the server when they arrive at or leave from a parking space, such status could become invalid after a certain period. Thus we switch segments' status back to Unknown when a timer expires. The timer length can be derived from statistic data or occupancy prediction [19] and can be adjusted through the observation of the crowdsourced data. Figure 2 shows all possible status changes and their causes. In addition to the occupancy status, we might also need to know the current capacity of the on-street parking lot in order to determine if we can navigate two cars to the same street at the same time.

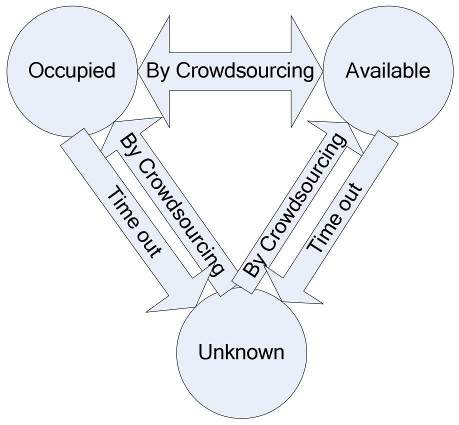


Figure 2. Status changes for road segments.

3.3 Data Collection Crowdsourcing

Data crowdsourcing for required information in a mobile environment could pose additional challenges to system design. An obvious problem is that the unsophisticated user interface and drivers' tight schedule require the operation to be simplified for the participants to answer questions. In the case of smart parking scenario, ideally, we might want smart parkers to observe the streets carefully and report a specific number for the parking capacity. However, most smart parkers could prefer to answering a much simpler Yes-No question by just pressing a button on their smart phones. Experiences from similar applications like Waze[6] show that a user-friendly interface and simple operations are key to recruit more contributors. In our solution, we allow the server to ask different questions as shown in table 1. We will study the difference between them in Section 4. Our study shows that the answer to a simple question is sufficient when the membership reaches a low percentage among all drivers.

Table 1. Different of questions smart parkers could be asked

#	*Question*	*Answers*	*Capacity*
Q1	How many parking spots on the street?	0,1,2,3…	As the answer
Q2	Any more parking spots on the street?	Yes/No	1(Yes)/0(No)
Q3	No question	No answer	Always 1

In addition to the requirement for a simplified operation, the limited vision of the participants could also keep the crowdsourcer from obtaining important data in the process of mobile crowdsourcing. For example, by answering the questions, smart parkers only inform the server of the situation of the street where they parked but tell nothing about the occupied streets they cruised through. In fact, we can infer such information through crowdsourced sensor data. More specifically, we assume a car to be cruising if it follows the server's instructions to reach a certain road segment but still keeps moving in a low speed. Then we consider the road segment as occupied where the car starts cruising. Furthermore, all streets that the car cruises through later without parking can also be regarded as occupied. In addition, we can mark a street as *Available* if some car leaves there. Since a car's cruising speed is only 20% of its normal driving speed, we can make the inferences above by just observing the sensor data like speed and location.

3.4 Parking Guidance

Once we get all the required information to annotate each street on the map with is dynamic status, the easiest way to do parking guidance is to make the locations of the free parking slots directlt available the smart parkers (as *OpenSpot* does). This way, all smart parkers consider the crowdsourcing system just as a bulletin board where they can read or write relevant information. They make their own decisions about where to park without the server's interference. In other words, the crowdsourcing process is not coordinated by a central entity.

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

Instead of letting drivers choose where to park by themselves, we propose a parking guidance algorithm on the server side to mitigate the problems mentioned above. 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. The value of the capacity can be obtained either from smart parkers' answers or by inference as indicated in Table 1. In order 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, we navigate the car to cruise toward streets with non-zero capacity. This way, we not only help the smart parker avoid unnecessary cruising but also increase the server's knowledge about Unknown streets. The detailed algorithm is shown in figure 3.

4. SIMULATION METHODOLOGY

4.1 Evaluation of Feasibility

Although we have enumerated a list of design options for building a crowdsourcing-based smart parking system in section 3, it is still unclear whether crowdsourcing will always be a feasible solution in such an application scenario, where every participant is supposed to contribute some data to the system. For simplicity, we assume that every driver has the same experience or knowledge about the parking spaces around their destination and those parking spots are equally open to everyone. Then we can use simulations to evaluate the feasibility from three aspects.

First, we need to investigate whether crowdsourcing can always make the smart parkers better off than ordinary drivers. In other words, we will evaluate different crowdsourcing strategies and compare the performance between the two groups of drivers. A feasible solution should guarantee that the majority of the smart parkers will spend less time in searching and will park closer to their destination than ordinary drivers, even when few smart parkers participate. Also, we want to evaluate the impact of different design options on the system's performance. For example, can a system which only asks simplified questions about parking availability, still guide smart parkers to the right place? Another factor we need to consider is the existence of free riders. If the system is vulnerable to free riders, strict access control or penalty mechanism would be necessary. On the other hand, if the fair parkers do not t notice an obvious degradation in the quality of service they receive, the tolerance to free riders would facilitate the growth of the crowdsourcing community.

```
If(Event==ParkingRequest)
    d=SmartParker.Destination

    S={s|s∈StreetsClosestTo(d),s.Capacity>0}

    SmartParker.StreetToPark=
        S.StreetClosestTo(SmartParker.CurrLocation)
    SmartParker.StreetToPark.Capacity--
If(Event==ParkingComplete)
    c=SmartParker.CurrStreet
    c.Status=SmartParker.ObserveStatus()
    c.AdjustCapacity()
    if(c!=SmartParker.StreetToPark)
        SmartParker.StreetToPark.AdjustCapacity()
If(Event==CarCruising)
    c=SmartParker.CurrStreet
    c.Status=Occupied, c.Capacity=0

    S={s|s∈c.AdjacentStreets,s.Capacity>0}
```

Figure 3. Parking guidance algorithm by coordinated crowdsourcing.

4.2 Simulation Environment

In order to simulate the crowdsourcing 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 take those crowdsourcing-related factors into account as we discussed in section 3. Since no existing solutions can satisfy both requirements, we modified an open source road traffic simulator, namely SUMO[21], to meet our needs.

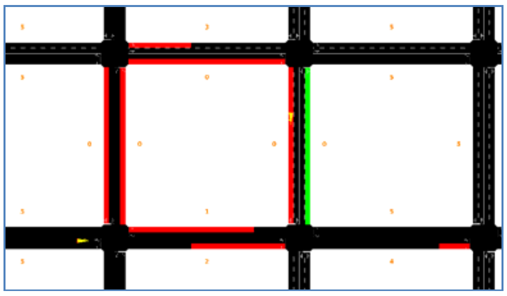


Figure 4. Simulated parking scenario by SUMO

SUMO is a microscopic road traffic simulator, which allows simulating thousands of single vehicles moving through a road network. It is capable of capturing the geospatial properties of each vehicle in motion like location and speed at any moment, which corresponds to our assumption about smart parkers, who should be able to report such information to the server. However, it heavily depends on predefined configuration files to determine the departure time and the route of each vehicle. In order 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, we add the parking capacity as a new property of 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, the data collection process and parking navigation is also implemented inside SUMO to reflect different crowdsourcing strategies mentioned in section 3. We use the length of red line to denote the occupancy of parking slots along a certain street on SUMO's map so that we can monitor how on-street parking spaces are consumed by a sequence of arriving vehicles. Figure 4 shows an area where cars cruise for an open spot.

4.3 Parameter setting

We try to evaluate the feasibility of the aforementioned crowdsourcing system in a simple but realistic scenario, where hundreds of vehicles are heading for the same destination during a certain short period and almost no car leaves the parking lots around the place at that time. This often happens around office buildings and park-and-ride facilities [22] during rush hours or at some stadium before a hot game kicks off. In this case, we can focus more on the impact of different design choices for the crowdsourcing system rather than statistics facts about parking lots usage around a certain area.

The road network in our simulations is defined to be a 1 km^2 square region, which is evenly divided into nine rows and nine columns of blocks by two-lane streets. In addition to the two traffic lanes, each street has an on-street parking lot, which is not drawn on the map but has a capacity as the roadside figure shows. The parking capacity is set to 5 for each side of the street. When a car parks on a street, the street's parking capacity decreases by one and the car is removed from the map. We suppose the block in the center is a common destination of all drivers and everyone tries to park close to it in order to reduce the walking distance. For each round of experiment, we generate a sequence of about 1,000 vehicles, which enters the map according to a Poisson process from one of the four corners toward the center.

Whenever a car is inserted into the map, its driver is configured to be either a smart parker or an ordinary driver according to a certain probability so that we can control the approximate ratio between these two groups of drivers. An ordinary driver picks one of the streets around the central block as his destination and drives at a normal speed. If the driver can't find an open spot on either side of the destination street, he will have to cruise around randomly until he can find one on some street else. The speed limit for normal driving is 50km/h while the cruising speed is below 10km/h.

When a smart parker moves close to the central block, the server will show him suggestions about the available parking place. Although there are different ways to do the parking navigation as we will explain later, we can't guarantee that the smart parker will definitely find an available slot on the suggested street considering the existence of ordinary drivers. For simplicity, we suppose that a smart parker also starts to cruise if he follows the server's suggestion but reaches a fully occupied on-street lot. However, the smart parker might be guided during cruising if we adopt certain crowdsourcing strategy. The speed limit for driving and cruising also applies to smart parkers.

5. EVALUATION

5.1 Uncoordinated vs. Coordinated Search

In the first set of experiments, we try to figure out if smart parkers can outperform ordinary drivers no matter what kind of crowd-sourcing strategy is applied. We consider the walking distance and the average cruising time as the two primary criteria when comparing smart parkers with ordinary drivers. We first assume that the system adopts a pure uncoordinated crowdsourcing strategy: that is each smart parker just follows a predecessor whose parking place is the closest one to the destination and not fully occupied yet. Figure 5 compares two groups of drivers with regard to average walking distance. The walking distance is measured in terms of blocks away from the destination and we collect the data as the membership of the smart parkers increases from 10% to 50% among all drivers. As the figure shows, the uncoordinated search for parking spots leads to longer walking distance for smart parkers. Since the system does not provide smart parkers with an integrated view around the region, they miss potential vacancies closer to the destination.

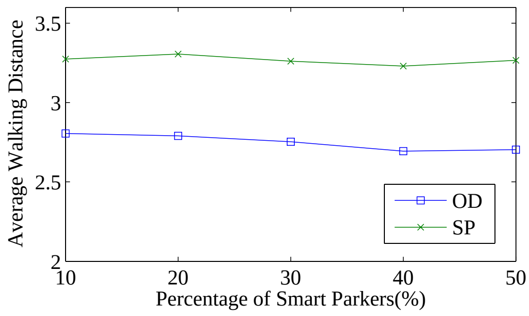


Figure 5. Performance comparison in uncoordinated crwodsourcing

As shown in the previous figure, the uncoordinated search approach (also used by Google's OpenSpot) fails to help smart parkers do a better job than ordinary drivers in the search of parking spots regardless of how many drivers participate. In contrast, the coordinated strategy we proposed in section 3 guides smart parkers more wisely. We assumed the system collects information by applying option Q1 in Table 1. In addition, it assigns smart parkers to explore unknown areas and helps them cruise more efficiently by avoiding occupied streets. As Figure 6 shows, such an approach achieves a lower average walking distance for smart parkers.

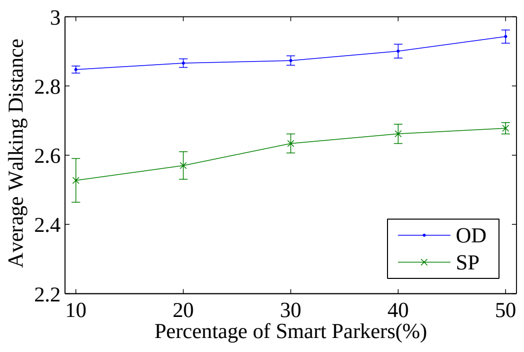


Figure 6. Performance comparison in coordinated crwodsourcing

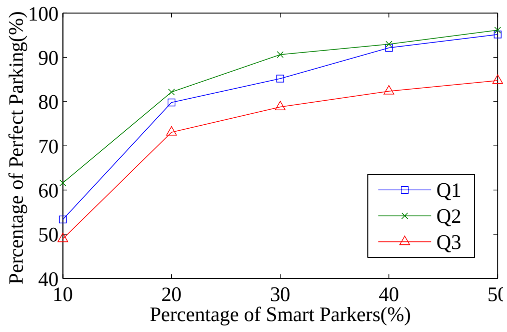


Figure 7. Influence of data accuracy to the performance

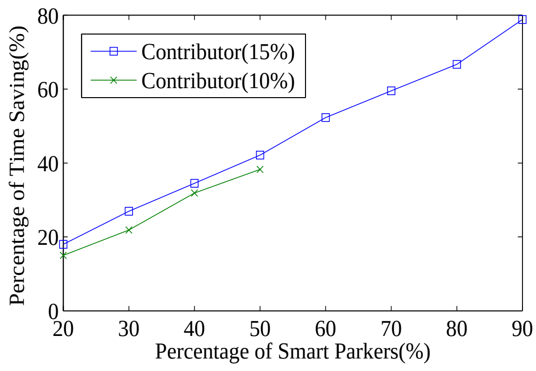


Figure 8. Increased time saving by free riders

5.2 Data Collection Accuracy: Evaluating the Impact of Various Design Options

While the coordinated search approach is capable of realizing the goal of smart parking in theory, some requirements in the system might not be practical under all circumstances. For example, drivers may not always have the time to count the exact number of available parking slots on the street where they park. In order to design a feasible solution for the real world, we need to evaluate the impact of different design options and figure out if a simplified system can still work.

We make all smart parkers answer the same question picked in table 1 to report relevant information to the server and then record the percentage of perfect parkers as the membership of smart parkers grows. Here perfect parkers denote those smart parkers who can find an open spot immediately by following the parking guidance. We plot the results in figure 7 with each line for a specific question. The figure reflects that the answers to a Yes-No question about the parking availability provide sufficient information for the server to implement a useful navigation service.

5.3 Dealing with Freeriders

According to the simulations results, the average cruising time of ordinary drivers doesn't change much as we adjust the quantity of smart parkers. This means the overall cruising time of all drivers would remain a constant if the smart parking system is not available. Although free riders could have a negative influence on the quality of the service, what they gain might be much more than others' loss. If we consider all drivers as a whole, the crowdsourcing system could boost social welfare by allowing more free riders to use the service, which means less time and fuel will be wasted in cruising, as long as the service is still usable. In the following simulations, we assume the contributors account for 10% or 15% among all drivers while the percentage of smart parkers grows from 20% to 90% of the 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 8 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 with only 10% of all drivers contributing to the system.

6. CONCLUSION

Mobile crowdsourcing is a viable mechanism for realizing applications that improve daily life. Mobile crowdsourcing, however, introduces new design tradeoffs that, usually, do not exist, in traditional web-based crowdsourcing initiatives. In this paper, we take smart parking as a case study and simulate this specific application scenario of mobile crowdsourcing. By comparing the simulation results for different design options and assumed situations, we shed light on the challenges and characteristics pertaining to the mobile crowdsourcing ecosystem. Meanwhile, we summarize some basic principles for a better practice of mobile crowdsourcing. In particular, our study suggest the following:

First, spontaneous crowdsourcing in a mobile environment can lead to herd behavior rather than collective intelligence as we can see in the traditional web-based crowdsourcing. Since each participant only has a limited view of his surroundings and the integrated picture of the physical world is not available, newcomers could be easily misled by predecessors and continue to lower the quality of the crowdsourced data. To deal with this issue, we propose the concept of coordinated search, in which the server integrates information from all participants, guides their search for parking spots, and encourages them to explore unknown area as well. Through simulations, we show that this approach is effective approach in mobile circumstances.

Second, the participation rate is much more important than the amount of information each individual contributes. As the results of a series of simulations show, when the membership of crowdsourcing participants passes a certain participation rate, the outcome does not change much, no matter whether each individual contributes accurate data or just approximate data. However, if the participation rate is not high enough, a sophisticated data collection mechanism becomes necessary to compensate the lack of data sources.

Finally, the crowdsourcing-based application might continue to increase social welfare by accepting more free riders, if we can maintain a moderate level of contribution among participants. In the context of mobile crowdsourcing, free riders could reduce the quality of the crowdsourcing-based service because they might change the status of the physical environment without reporting their behavior. However, the overall benefit of all participants could still rise significantly in contrast with the slightly degraded service quality as long as a certain percentage of the members keep contributing.

7. REFERENCES

[1] D.C. Brabham, T.W. Sanchez and K. Bartholomew, "Crowdsourcing Public Participation in Transit Planning: Preliminary Results from Next Stop Design Case," in TRB 89th Annual Meeting Compendium of Papers DVD, 2010.

[2] A. Sorokin and D. Forsyth, "Utility data annotation with Amazon Mechanical Turk," in Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on, pp. 1-8, 2008.

[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, Baik Hoh, Deepak Ganesan, Ken Tracton, Toch Iwuchukwu, and Juong-Sik Lee., "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/, .

[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, Daqing Zhang, Lin Sun, Chao Chen, Shijian Li, Guande Qi and Qiang Yang, "Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset," in Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on, pp. 63-68, 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 12th IEEE International Conference on, pp. 3-6, 2011.

[12] S. Reddy, A. Parker, J. Hyman, J. Burke, D. Estin and M. Hansen, "Image Browsing, Processing and Clustering for Participatory Sensing: Lessons from a DietSense Prototype," in Proceedings of the Workshop on Embedded Networked Sensors (EmNetS), Cork, Ireland, June 2007, .

[13] M. Mun, S. Reddy, et al, "PEIR, the Personal Environmental Impact Report, as a Platform for Participatory Sensing Systems Research," in Proceedings of ACM MobiSys, Krakow, Poland, June 2009, .

[14] E. Miluzzo, N. Lane, K. Fodor, R. Peterson, S. Eisenman, H. Lu, M. Musolesi, X. Zheng, A. Campbell, "Sensing Meets Mobile Social Networks: The Design, Implementation and Evaluation of the CenceMe Application," in Proceedings of ACM SenSys, Raleigh, NC, USA, November 2008. .

[15] Jatuporn Chinrungrueng, Udomporn Sunantachaikul and Satien Triamlumlerd, "Smart Parking: An Application of Optical Wireless Sensor Network," in Applications and the Internet Workshops, 2007. SAINT Workshops 2007. International Symposium on, pp. 66-66, 2007.

[16] Rongxing Lu, Xiaodong Lin, Haojin Zhu and Xuemin Shen, "SPARK: A New VANET-Based Smart Parking Scheme for Large Parking Lots," in INFOCOM 2009, IEEE, pp. 1413-1421, 2009.

[17] Yanfeng Geng and C.G. Cassandras, "A new “smart parking” system based on optimal resource allocation and reservations," in Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on, pp. 979-984, 2011.

[18] SFMTA, "SFPark- About the Project," Http://sfpark.org/about-the-Project/, .

[19] M. Caliskan, A. Barthels, B. Scheuermann and M. Mauve, "Predicting Parking Lot Occupancy in Vehicular Ad Hoc Networks," in Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th, pp. 277-281, 2007.

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

[21] Michael Behrisch, Laura Bieker, Jakob Erdmann and Daniel Krajzewicz., "SUMO - Simulation of Urban MObility: An Overview," in SIMUL 2011, The Third International Conference on Advances in System Simulation, 2011.

[22] Flavia W. K. Tsang, Amer S. Shalaby and Eric J. Miller, "Improved modeling of park-and-ride transfer time: Capturing the within-day dynamics," Journal of Advanced Transportation, vol. 39, pp. 117-137, 2005.