sensors

MDPI

Article

Storage Management Strategy in Mobile Phones for Photo Crowdsensing

En Wang [1], Zhengdao Qu [2], Xinyao Liang [3], Xiangyu Meng [1,*], Yongjian Yang [1], Dawei Li [4] and Weibin Meng [5]

1 Department of Computer Science and Technology, Jilin University, Changchun 130012, China; wangen@jlu.edu.cn (E.W.); yyj@jlu.edu.cn (Y.Y.)
2 Department of Software, Jilin University, Changchun 130012, China; quzd2217@mails.jlu.edu.cn
3 College of Business Administration, Anhui Finance and Economics University, Anhui 233030 China; 20181332@aufe.edu.cn
4 Department of Computer Science, Montclair State University, Montclair, NJ 07043, USA; dawei.li@montclair.edu
5 Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China; mwb16@mails.tsinghua.edu.cn
* Correspondence: xymeng512@jlu.edu.cn

Received: 13 February 2020; Accepted: 14 March 2020; Published: 13 April 2020

Abstract: In mobile crowdsensing, some users jointly finish a sensing task through the sensors equipped in their intelligent terminals. In particular, the photo crowdsensing based on Mobile Edge Computing (MEC) collects pictures for some specific targets or events and uploads them to nearby edge servers, which leads to richer data content and more efficient data storage compared with the common mobile crowdsensing; hence, it has attracted an important amount of attention recently. However, the mobile users prefer uploading the photos through Wifi APs (PoIs) rather than cellular networks. Therefore, photos stored in mobile phones are exchanged among users, in order to quickly upload them to the PoIs, which are actually the edge services. In this paper, we propose a utility-based Storage Management strategy in mobile phones for Photo Crowdsensing (SMPC), which makes a sending/deleting decision on a user's device for either maximizing photo delivery ratio (SMPC-R) or minimizing average delay (SMPC-D). The decision is made according to the photo's utility, which is calculated by measuring the impact of reproducing or deleting a photo on the above performance goals. We have done simulations based on the random-waypoint model and three real traces: *roma/taxi*, *epfl*, and *geolife*. The results show that, compared with other storage management strategies, SMPC-R gets the highest delivery ratio and SMPC-D achieves the lowest average delay.

Keywords: mobile crowdsensing; edge computing; photo crowdsensing; storage management; utility-based

1. Introduction

Recent years have witnessed the obvious growth of mobile devices, where all kinds of sensors (e.g., locator, camera, accelerometer) are implanted to make the mobile devices have a powerful sensing ability. For this reason, a mobility-based crowdsourcing way called *mobile crowdsensing* is presented [1] to gather a group of mobile users for collectively finishing a common sensing task using their mobile devices. Particularly, a novel case of mobile crowdsensing is collecting graphical information through the cameras of users' mobile phones [2,3].

For example, in a disaster area or a battle field network, the control center requires the pictures of the specific targets. The mobile devices of the people around the targets could be used to take pictures

and the holders could upload the photos to the control center. Another case is that Google Maps services provide customers with a visual feeling of the specific places, through the street cameras. Obviously, a picture could give the customers richer content than text description. A well-known application Waze [4], [5] utilizes the thought of crowdsensing to provide the timely road traffic situation. Drivers could share the road traffic information through taking photos for the accidents, dangers, and also road conditions; then, the sensing data could be efficiently uploaded to the control center and translated into the related knowledge, which could serve back for the drivers.

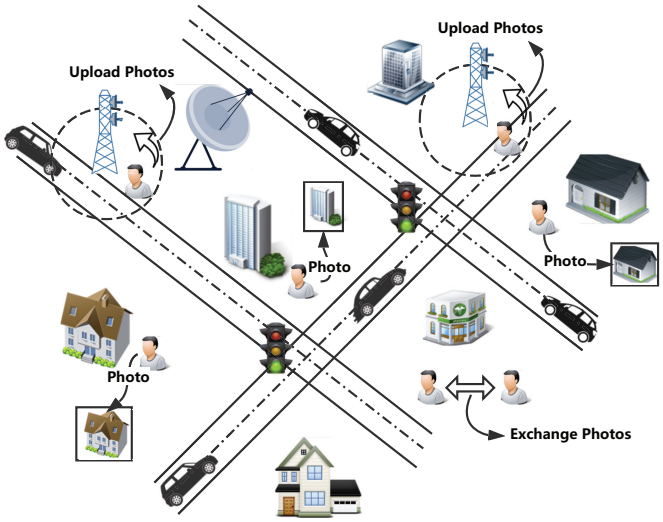


Figure 1. A running example of photo crowdsensing.

There are some works focusing on mobile crowdsensing including system design [6–8], worker recruitment strategies [9–13], incentive mechanisms [14–19], and so on. However, there have been a small amount of works focusing on mobile photo crowdsensing. More importantly, an important issue is the storage problem in mobile photo crowdsensing because the storage space for a phone to serve for the crowdsensing application is usually limited; even the total storage space is enough (most space is used by the other applications in user's daily life). As shown in Figure 1, users move around an area to cooperatively perform a mobile crowdsensing task. There are some Wifi APs regarded as the PoIs, which are actually the edge services in Mobile Edge Computing (MEC) [20]. When they see the targets (buildings or sights), they take photos and store the photos in their storage. Users need to upload the photos to the server in order to finish the photo crowdsensing task. They prefer uploading the photos through PoIs freely rather than uploading through a costly cellular network [21]. Therefore, they copy the photos when they enter the communication area of each other, in order to upload the photos as soon as possible. However, each mobile phone has a limited amount of storage that serves for crowdsensing tasks. In order to efficiently finish the photo crowdsensing task, users need to decide which photo to copy first when the connection duration among users is limited. Moreover, they also need to decide which photo to delete first when the storage of the mobile phone is overflowed. Of course, the final purpose is to upload as many photos as possible; then, the crowdsensing task could be well finished.

This paper studies a storage management framework, which consists of a parameter collection module, utility calculation module, and decision module in mobile phones for photo crowdsensing. The purpose of this work is to make a smart decision about which photo should be sent or deleted. The smart decision (decision module) is based on measuring the increase or decrease for every photo copy on either the photo delivery ratio or average delivery delay. The impact is quantified as the photo's utility (utility calculation module), which is achieved by corresponding ordinary differential equations (ODEs). The ODE solution calculates each photo's utility value, according to the collection of the following states: the number of photo copies, and the number of users that have ever stored the photo

copy (parameter collection module). Finally, we propose a utility-based Storage Management strategy in mobile phones for Photo Crowdsensing (SMPC). We have done the simulations, which prove that the proposed storage management strategy achieves a good performance.

The main contributions of this paper are briefly summarized as follows:

- We propose a storage management framework, which includes the parameter collection module, utility calculation module, and decision module to address the photo crowdsensing problem in mobile phones.
- We propose two utility-based Storage Management strategies in mobile phones for Photo Crowdsensing (SMPC), one is for maximizing delivery ratio (SMPC-R), and the other one is for minimizing average delay (SMPC-D).

The remainder sections are organized as follows: In Section 2, we describe the process to formulate the question to mathematical linguistics. The storage management framework is presented in Section 3. In Section 4, we propose two utility-based storage management strategies in mobile phones for photo crowdsensing, for the goals of delivery ratio and average deliver delay, respectively. In Section 5, we test the performance of the storage management strategies in this paper through many groups of simulations. We review the related work in Section 6 and conclude the paper in Section 7.

2. Related Work

There are two parts of existing research related to the work in this paper: buffer management in DTNs [22,23] and photo crowdsourcing.

2.1. Buffer Management in DTNs

Elwhishi et al. [24] present a new framework to schedule message, thus increasing the forwarding radio and lowering the propagation delay. Wang et al. [25] propose a strategy to dispatch or drop the message, in order to make full use of the network resource and reduce the buffer is overflowed. Krifa et al. [26] present an efficient buffer control strategy about the network using the theory of collision-based message propagation. Based on the result of [26], Krifa et al. [27] make progress and put forward a joint scheduling and drop strategy to optimize the metrics; furthermore, they propose a distributed algorithm to approximate the optimal algorithm. Moreover, Krifa et al. [28] utilize collected statistics to estimate all useful parameters, in order to enhance their previous work. Aruna et al. [29] regard the DTN routing as a resource allocation problem, then propose a DTN routing protocol to optimize significant performances such as the worst transmission delay and so on. Matzakos et al. [30] consider both the delivery probability and delay requirements in DTN and propose a distributed buffer management approach based on the heterogeneous contact rates and sparse contact graphs.

The above research focuses on the buffer-management of nodes in DTNs. The works are similar to this paper. However, the strategies proposed in DTNs could not be directly used in mobile phones for photo crowdsensing.

2.2. Photo Crowdsourcing

Zhou et al. [31] propose an urban traffic monitoring system by using bus riders' mobile phones which can turn buses into probes for monitoring buses' travel situations and inferring instant traffic map of the city. Rula et al. [32] propose an approach which leverages the built-in incentives of location-based gaming and social applications to exert limited control over the actions of participators. Siahaan et al. [33] study the influence of different experimental settings and rating scales on the reliability and repeatability of aesthetic appeal assessments in the process of collecting picture aesthetic appeal ground truth data by a systematic way. Wang et al. [34] propose a crowdsourcing based scene reporting system (RVShare). When the real-time picture of a location is requested, the RVShare system could provide the picture by means of the person who is passing by the location. Moreover, in order to perfect the system, they also design a task allocation scheme, a reward scheme, and a processing

flow to deal with the pictures uploaded. Considering the security of the data such as photos, Wu et al., propose a data protection and recovery method based on the incentive [35] and a prediction-based risk defense strategy [36] for the data. Zhuo et al. [37] propose a tripartite architecture for data aggregation and analysis to mitigate the pressure of resource constrained requesters in mobile crowdsourcing by means of cloud support; they achieve privacy-preserving of data and identity of workers. The requester is able to verify the results received from the cloud. Wu et al. [38] propose a picture crowdsourcing framework through disruption tolerant networks (DTNs) as well as a picture selection algorithm, which uses the photo meta data to measure the priority of photos and takes the coverage of overlap photos into consideration. Wu et al. [39] design a pricing incentive mechanism based on quality of videos named Vbargain for the purpose of encouraging mobile users to cooperate to deliver video data effectively and easing the burden of mobile data traffic. They take the marginal gains of video quality and its duplicate into consideration and view the process of data delivery as a behavior of market to stimulate the workers. Zhou et al. [40] focus on selecting a proper photo subset of an area from the server to a requester. Based on the coverage on areas and quality on the views, they propose a photo selection approach that contains two schemes: basic and PoI number-aware photo selection schemes.

The above works focus on utilizing the photo crowdsourcing to efficiently finish the crowsensing task. However, as far as we know, there is no work focusing on storage management in mobile phones for photo crowdsensing. Above all, this paper is the first work to propose a storage management strategy in mobile phones for photo crowdsensing.

3. Network Model and Problem Formulation

3.1. Network Model

We pay attention to a mobile network for photo crowdsensing including N mobile users, which will randomly take photos (the requesters publish some tasks to take the photos for the specific targets) through their mobile phones. The photos need to be uploaded before a deadline. We assume that all the photos have a uniform time to live (TTL). To minimize total uploading cost of all the photos, users prefer uploading the photos through a free-Wifi connection, which is defined as a PoI, rather than uploading them costly. In all the PoIs, users could upload all the photos in the storage. Each user has a uniform communication range d, when a pair of users enter the communication range of each other, they could exchange some photos, whose number is decided by their contact time. A longer contact time leads to a larger number of exchanged photos. The exchange process is an epidemic [41] copy process, which utilizes all the transmission probabilities to copy the photos to all the encounters. Here, we assume that some incentive mechanisms have been designed to make users prefer assisting with finishing the MCS tasks, while the incentive work is not considered in this paper. A photo could just be deleted in the following two situations: successfully uploaded to a PoI, or deleted by the management strategy when the storage is overflowed. Before uploading to a PoI, the photos need to be stored in a mobile phone's storage. We assume that every mobile phone has a uniform storage space, and all the photos are also the same size. When the storage is overflowed, some photos should be deleted from the storage. The network environment is shown in Figure 2. The main symbols are listed in Table 1.

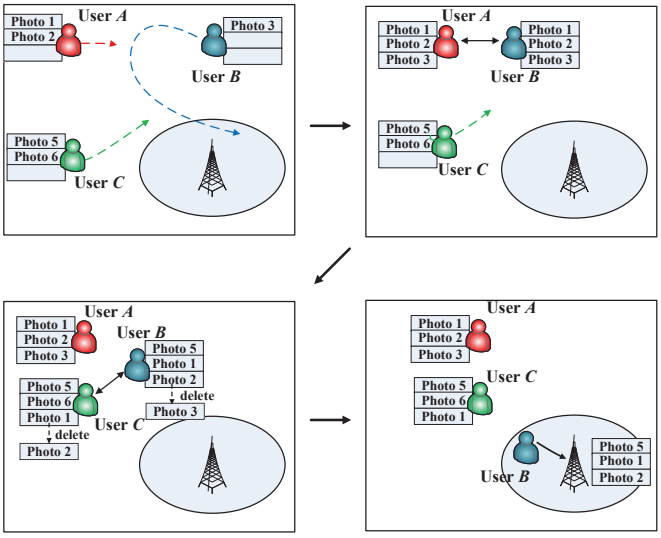


Figure 2. A running example of the storage management problem in mobile phones for photo crowdsensing. User A with photos 1 and 2 encounters user B with photos 3; they exchange photos in the epidemic manner. After a period of time, user B encounters user C; they exchange photos and delete the photo with the lowest utility when the storage is overflowed. Finally, when user B enters a PoI, it uploads all the photos successfully.

Table 1. Main Symbols of this paper

Symbol	Meaning
N	the number of users minus one
TTL_i	uploaded deadline for the sensing photo i
r_i	remaining live time for photo i
t_i	elapsed time for photo i from its generation time to the current time ($t_i = TTL_i - r_i$)
$n_i(t_i)$	copy number of photo i in the storages when the time is t_i
$d_i(t_i)$	copy number of deleted photo i when the time is t_i
$m_i(t_i)$	number of devices ever carry photo i when the time is t_i
$u(t_i)$	number of users that have ever been in any PoI in photo i's elapsed time t_i
λ_1	variable for intermeeting time distribution among users
E_1	expected intermeeting time between users ($E_1 = \frac{1}{\lambda_1}$)
λ_2	variable for intermeeting time distribution of user and PoI
E_2	expected intermeeting time between user and PoI ($E_2 = \frac{1}{\lambda_2}$)
U_i	utility of photo i
P_{t_i}	probability to finish delivering photo i to PoIs when t_i
P_{r_i}	probability to finish delivering the undelivered photo i when the remaining time is r_i
P_i	probability to finish delivering photo i
D_{t_i}	expected delivery delay for photo i after elapsed time t_i
D_{r_i}	expected delivery delay for photo i within remaining time r_i
D_i	delivery delay for photo i

3.2. Problem Formulation

As described in the network model, in order to finish the photo crowdsensing, users take photos when they see some interesting activities or sights. In addition, they exchange photos in the epidemic manner, in order to deliver the photo to a PoI. In this way, users cooperatively finish a phone crowdsensing task through their mobile phones. However, the epidemic exchanging strategy consumes a large number of storage resources, and users' mobile phones also have a limited storage space, which could not be regarded as an abundant resource. Therefore, in order to achieve a good photo crowdsensing result, we should propose a storage management strategy in mobile phones.

Through the above descriptions, when a user takes a photo in its storage, we could regard the user as source and consider the PoIs as destinations. The purpose of photo crowdsensing is to deliver the photos from sources to destinations with the help of the other users. However, taking the limited storage space of mobile phones into consideration, our work mainly solves the next two problems: (1) when one or more photos are stored in a user's phone storage and the user could not decide whether the connection could be enough to copy these photos. In order to achieve the best photo crowdsensing performance (deliver ratio or average delay), we should make a decision about which photo to be with a higher priority. (2) If a new photo comes to a user's storage full of photos, in order to achieve the best photo crowdsensing performance, we must make a decision about which photo to delete between the persistent photos and the new incoming photo.

Here, we present a storage management strategy in mobile phones for photo crowdsensing, which first expresses the photo crowdsensing performances (delivery ratio and average delay) as a function of variables. The utility for each photo is achieved through the derivative value of the photo crowdsensing performance. If the transmission speed for the connections is not enough to send all the photos stored in the storage, the user will copy photos from high to low for the utility of each photo. If storage is overflowed, the user will make a decision about which photo to delete according to the utility, in order to achieve the best photo crowdsensing performance.

We first define several notations. For example, we regard the delivery ratio as the photo crowdsensing performance (average delay has a similar derivation). P_i is the probability to finish delivering photo i, n_i is the number of photo i in users' storages when the time is t_i. U_i is the utility of photo i measured by the impact of sending or deleting the photo on the performance. Then, the following situations are considered:

$$\begin{cases} \Delta(n_i) = 1 & \text{copy photo } i. \\ \Delta(n_i) = 0 & \text{do nothing.} \\ \Delta(n_i) = -1 & \text{delete existing photo } i. \end{cases}$$

Therefore, $\Delta P_i = U_i \Delta(n_i)$, $U_i = \frac{\partial P_i}{\partial n_i}$.

According to the above descriptions, we schedule the photos in the photo's decreasing order, and photos in high order are copied when there is communication opportunity. If the storage is overflowed, then we will delete the photos with the lowest utility.

4. Proposed Storage Management Framework

Figure 3 uses a readily comprehensible view to describe the storage management framework of mobile photo crowdsensing, including three modules and transitions among them. The parameter collection module is used to collect the following three parameters $n_i(t_i)$, $m_i(t_i)$, and $u(t_i)$, which could be regarded as the inputs of the utility calculation module. The utility calculation module is the main part of the framework. The decision of sending or deleting the photos in storage is achieved according to the storage occupancy status and the utility value of the photos. Then, the three modules are described in detail as follows.

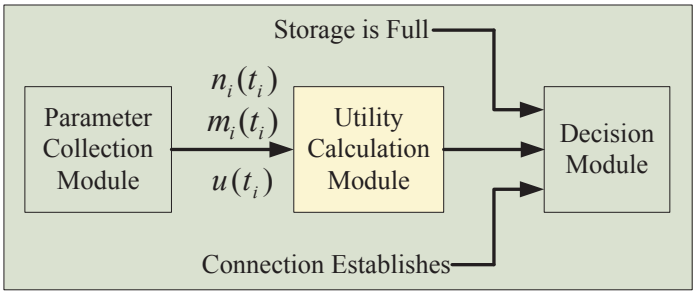


Figure 3. The storage management framework.

4.1. Parameter Collection Module

In order to calculate the per-photo utility, which is used to decide the sending and deleting priority, the network parameters are exchanged among users, which include the following data: (1) $n_i(t_i)$, which is the number of photo i when the time is t_i. (2) $m_i(t_i)$, which is the number of devices for every carry photo i when the time is t_i. (3) $u(t_i)$, which is the number of users that have been in any PoI before the photo i's elapsed time t_i. The exchange process ensures that each user could estimate the above three parameters through collecting and exchanging the information of the other users. Section 4 describes in detail how the parameters are collected.

4.2. Utility Calculation Module

In this subsection, we attempt to calculate the photo's utility; in other words, the following problem is addressed: $m_i(t_i)$, $n_i(t_i)$, and $u(t_i)$, and limited storage space for supporting epidemic photo exchanging strategy are known, we should find a suitable strategy about whether to receive the new photo or just reject the coming photo, in order to optimize our performance goals. In Section 4, we introduce in detail the process to calculate the photo's utility. Two optimization goals are regarded as the purposes in terms of photo crowdsensing.

4.3. Sending and Deleting Decisions

With the per-photo utility, the user first sorts the photos of its storage from high to low. When the storage is overflowed, the photo of lower utility could be first deleted from the storage; when there is a communication probability, the photos of higher utility could be copied first to an encounter. Figure 4 describes the sending and deleting processes: if the photo j's utility U_j in user A's storage is the highest. In addition, it is higher than U_B of photo i at user B, which is the lowest in user B. Then, photo i is deleted and a copy of photo j enters user B's storage when users A and B encounter each other. The detailed decision algorithm is shown in Algorithm 1, called Storage Management for Photo Crowdsensing (SMPC).

Algorithm 1 SMPC

Input:
Capacity of photos in a user's storage: n
ID for a new arriving photo: m
Output:
Sending photo: ID_S, Deleting photo: ID_D

```
for i =1 to n do
    Calculate U_i
Sort U_i in a progressive increase manner
Search for the highest U_h, and assign h to ID_S
Search for the lowest U_l, and assign l to ID_D
if there is a connection then
    return ID_S
if storage is overflowed then
    Calculate U_m
    if U_m < U_l then
        assign m to ID_D
    return ID_D
```

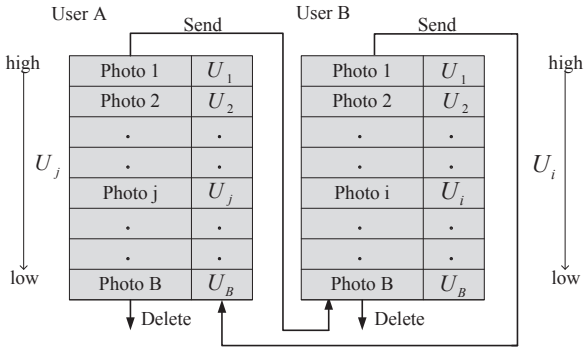


Figure 4. The management process in the user's storage.

5. Utility-Based Storage Management Strategy

5.1. Mobility Pattern in Mobile Crowdsensing

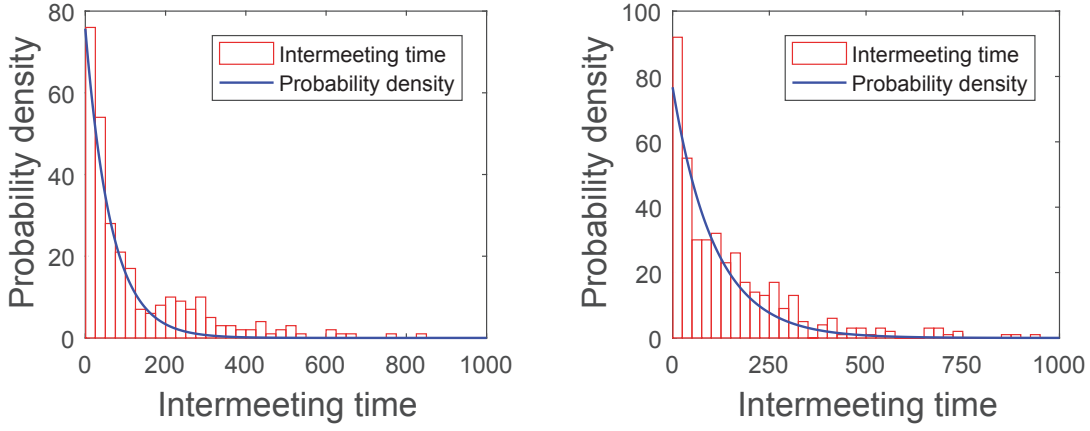
In the photo crowdsensing environment, users mainly utilize occasional communication opportunities to copy photos. Therefore, the intermeeting time among users and that between users and PoIs will make an important impact on delivery ratio and average delay of photos. Focusing on this, we first give the following two definitions:

Definition 1. *Intermeeting time E_1 focuses on a pair of users; it defines the time span between the last connection and the current connection.*

Definition 2. *Intermeeting time E_2 focuses on a PoI and a user; it also defines the time span between the last connection and the current connection.*

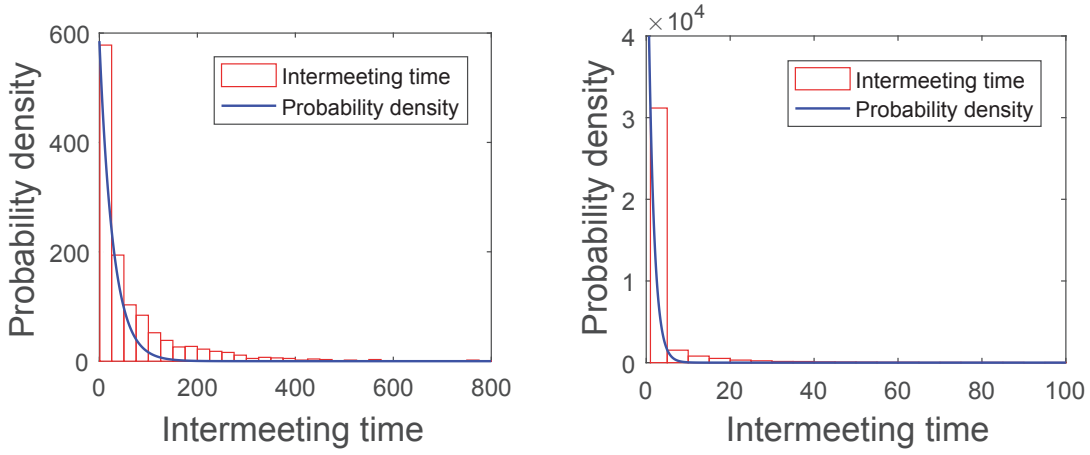
Recent research [42] proves that the intermeeting time satisfies an exponential distribution in some mobility patterns, such as random walk and random-waypoint because the simulations of this paper are tested in the random-waypoint scenario and three real-world traces (*roma/taxi trace set*, *epfl trace set*, and *geolife trace set*). The first group of simulations are about the distribution of the intermeeting times for the four scenarios above, in order to see if they can match the exponential distribution.

As shown in Figures 5–8, the intermeeting times approximately match the exponential distribution for the above situations: $f(x) = \lambda e^{-\lambda x}$ $(x \geq 0)$. λ_1 and λ_2 are shown in Table 1; then, we have $\lambda_1 = \frac{1}{E_1}$ and $\lambda_2 = \frac{1}{E_2}$.



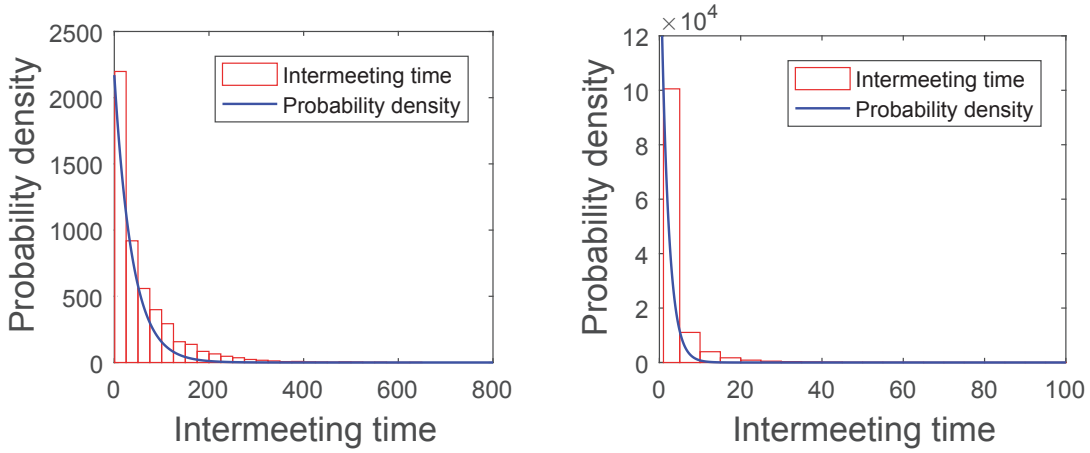
(**a**) Between users and PoIs (**b**) Among users

Figure 5. Random-waypoint mobility pattern.



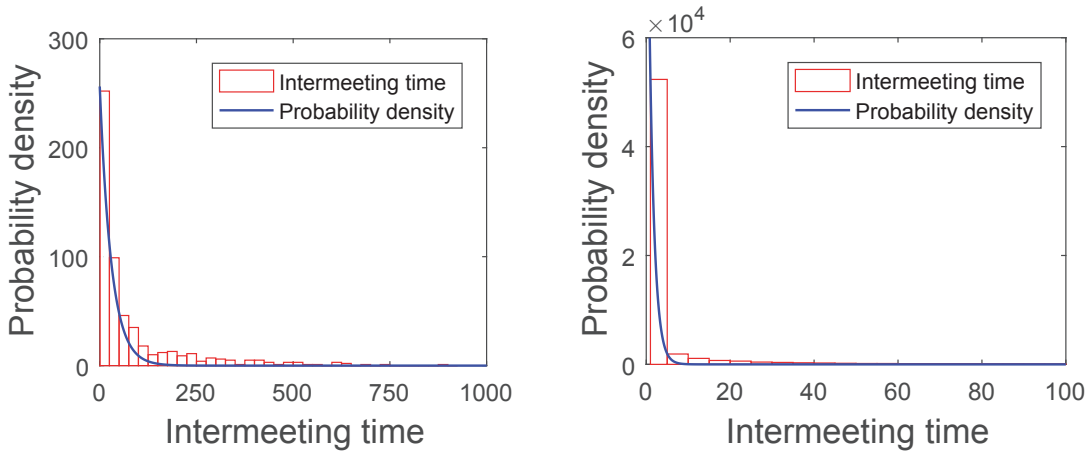
(**a**) Between users and PoIs (**b**) Among users

Figure 6. Roma/taxi trace set.



(**a**) Between users and PoIs (**b**) Among users

Figure 7. Epfl trace set.



(**a**) Between users and PoIs (**b**) Among users

Figure 8. Geolife trace set.

5.2. Maximization of Delivery Ratio

The probability to finish delivering photo i: P_i could be calculated through P_{t_i}, which is defined as probability to finish delivering photo i to PoIs when the time is t_i, and P_{r_i}, which is defined as probability to finish delivering the undelivered photo i when the remaining time is r_i. Then, P_i could be calculated as follows:

$$P_i = (1 - P_{t_i})P_{r_i} + P_{t_i} \tag{1}$$

The elapsed time for photo i since its born date is t_i. At time t_i, we assume that $m_i(t_i)$ users (excluding source) have carried a copy of photo i. Moreover, $u(t_i)$ represents the users that have ever been in any PoI before the current time t_i. The delivery ratio could be regarded as the probability that at least one user is not only among $m_i(t_i)$ users but also in $u(t_i)$ users. Therefore, the probability $P(t_i)$ can be shown as Equation (2):

$$P_{t_i} = 1 - \frac{C_N^{u(t_i)} C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{u(t_i)} C_N^{m_i(t_i)}} = 1 - \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} = 1 - \frac{(N-u(t_i))(N-u(t_i)-1)\cdots(N-u(t_i)-(m_i(t_i)-1))}{N(N-1)\cdots(N-(m_i(t_i)-1))} \tag{2}$$

Equation (2) is reasonable because the following two characters are achieved: (1) when $m_i(t_i)$ increases, $P(t_i)$ also increases, which is shown in Equation (3). This matches our common sense: more users that have seen photo i will lead to a higher delivery ratio at time t_i. (2) when $u(t_i)$ increases, $P(t_i)$ also increases, which is shown in Equation (4). This also makes sense because more users that have ever been in any PoI before time t_i will also lead a higher delivery ratio. Therefore, Equation (2) could precisely reflect the delivery probability of photo i:

$$\frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} > \frac{C_{N-u(t_i)}^{m_i(t_i)+1}}{C_N^{m_i(t_i)+1}} \tag{3}$$

$$\frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} > \frac{C_{N-u(t_i)-1}^{m_i(t_i)}}{C_N^{m_i(t_i)}} \tag{4}$$

P_{r_i} as shown in Equation (1) has been described in Table 1. The calculation of P_{r_i} is the major difficulty of the utility calculation model. First of all, we pay attention to the relationship between $n_i(t)$ and the time t. Taking the ordinary-differential-equation model [29] into consideration, we achieve the following:

$$\frac{\mathrm{d}n_i(t)}{dt} = \lambda_1 n_i(t)[N - n_i(t)], \tag{5}$$

where the parameter $\lambda_1 = \frac{1}{E_1}$. Through solving Equation (5), we achieve Equation (6) as follows:

$$n_i(t) = \frac{N n_i(0)}{n_i(0) + [N - n_i(0)]e^{-\lambda_1 N t}}. \tag{6}$$

Then, when the time is $t_i + r_i$, the number of users could be calculated as follows:

$$n_i(t_i + r_i) = \frac{N n_i(t_i)}{n_i(t_i) + [N - n_i(t_i)]e^{-\lambda_1 N r_i}} \tag{7}$$

Next, we attempt to calculate P_{r_i}. The equation $1 - P_{r_i}$ is the situation that photo i does not finish delivering when the time is t_i, and can not finish delivering in the rest of TTL: r_i ($r_i = TTL - t_i$). In other words, $1 - P_{r_i}$ is the situation in which not only can the $n_i(r_i)$ users with photo i not be the

destination PoI during r_i, but also the new users with photo i can not deliver the photo to any PoI before r_i. Then, the following equation is achieved:

$$1 - P_{r_i} = e^{-\lambda_2 n_i(t_i) r_i} \prod_{t=0}^{r_i} e^{-\lambda_2 n_i'(t_i+t)(r_i-t)} = \frac{e^{-\lambda_2 n_i(t_i) r_i} \prod_{t=0}^{r_i} e^{-\lambda_2 n_i'(t_i+t)(r_i)}}{\prod_{t=0}^{r_i} e^{-\lambda_2 n_i'(t_i+t)(t)}} = \frac{e^{-\lambda_2 n_i(t_i+r_i) r_i}}{e^{-\lambda_2 \int_0^{r_i} n_i'(t_i+t) t \mathrm{d}t}} \quad (8)$$

The quantity $n_i(t_i + r_i)$ in Equation (8) can be achieved as follows:

$$\begin{aligned} \int_0^{r_i} n_i'(t_i+t) t \mathrm{d}t &= \int_0^{r_i} t \mathrm{d} n_i(t_i+t) = \Big|_0^{r_i} t n_i(t_i+t) - \int_0^{r_i} n_i(t_i+t) \mathrm{d}t \\ &= r_i n_i(t_i+r_i) - (N r_i + \frac{\ln[n_i(t_i) - n_i(t_i) e^{-\lambda_1 N r_i} + N e^{-\lambda_1 N r_i}]}{\lambda_1}) + \frac{\ln(N)}{\lambda_1} \end{aligned} \quad (9)$$

By combining Equations (7)–(9), the solution of P_{r_i} is easy to achieve:

$$1 - P_{r_i} = \frac{N^{\frac{\lambda_2}{\lambda_1}}}{e^{-\lambda_2 N r_i} [n_i(t_i) - n_i(t_i) e^{-\lambda_1 N r_i} + N e^{-\lambda_1 N r_i}]^{\frac{\lambda_2}{\lambda_1}}} \quad (10)$$

Then, the final equation P_i is shown as follows:

$$P_i = 1 - \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} + \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} (1 - \frac{N^{\frac{\lambda_2}{\lambda_1}}}{e^{-\lambda_2 N r_i} [n_i(t_i) - n_i(t_i) e^{-\lambda_1 N r_i} + N e^{-\lambda_1 N r_i}]^{\frac{\lambda_2}{\lambda_1}}}) \quad (11)$$

Then, the utility for photo i can be derived through the following equation:

$$\Delta(P_i) = \frac{\mathrm{d}(1 - P_{t_i}) P_{r_i}}{\mathrm{d} n_i(t_i)} = \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} \frac{\lambda_2}{\lambda_1} (1 - e^{-\lambda_1 N r_i}) \frac{N^{\frac{\lambda_2}{\lambda_1}}}{e^{-\lambda_2 N r_i} [n_i(t_i) - n_i(t_i) e^{-\lambda_1 N r_i} + N e^{-\lambda_1 N r_i}]^{\frac{\lambda_2}{\lambda_1}+1}} \Delta n_i(t_i) \quad (12)$$

The storage management proposed here is to optimize the delivery ratio performance in the photo crowdsensing. If a photo i is copied successfully, the photo i's copies add one [$\Delta n_i(t_i) = +1$]; if nothing is done for photo i, the number of photo i's copies does not change [$\Delta n_i(t_i) = 0$]; if photo i is deleted from a user's storage, the number of photo i subtracts one [$\Delta n_i(t_i) = -1$]. Therefore, the utility of photo i is just $\Delta(P_i)$. The delivery ratio per-photo utility is achieved:

$$U_i = \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} \frac{\lambda_2}{\lambda_1} (1 - e^{-\lambda_1 N r_i}) \frac{N^{\frac{\lambda_2}{\lambda_1}}}{e^{-\lambda_2 N r_i} [n_i(t_i) - n_i(t_i) e^{-\lambda_1 N r_i} + N e^{-\lambda_1 N r_i}]^{\frac{\lambda_2}{\lambda_1}+1}}. \quad (13)$$

5.3. Minimization of Average Delay

In this section, focusing on the performance of average delay for time-sensitive photo crowdsensing [43], we assume that the deadlines for all the photos are so long that a close-to-1 delivery ratio could be achieved. The following derivation gives the way to optimize the performance of the average delay for all the photos.

Delivery delay for photo i is defined as D_i. If photo i has been successfully transmitted to a PoI at time t_i, the expected delivery delay for photo i is defined as D_{t_i}. Otherwise, the expected delivery delay for photo i within the TTL is D_{r_i}:

$$D_i = P_{t_i} D_{t_i} + (1 - P_{t_i}) D_{r_i} \tag{14}$$

It is not difficult to find that, D_{t_i} could be regarded as 0 if the photo i has already been delivered. The earliest time for the first photo i to be delivered to PoI is an exponential distribution with average value: $\frac{1}{n_i(t_i)\lambda_2}$. Thus, $D_{r_i} = t_i + \frac{1}{n_i(t_i)\lambda_2}$. Then, we could achieve the expected average delay (D_i),

$$D_i = (1 - \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}}) * 0 + \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} (t_i + \frac{1}{n_i(t_i)\lambda_2}) \tag{15}$$

Now, we differentiate D_i in terms of $n_i(t_i)$ to propose a strategy that maximizes the per-size increase in D_i,

$$\Delta(D_i) = -\frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} \frac{1}{n_i(t_i)^2 \lambda_2} \Delta n_i(t_i) \tag{16}$$

The best deleting or sending decision will be the one that maximizes $|\Delta(D_i)|$. We obtain the following equation for calculating per-photo average delay utility:

$$U_i = \frac{C_{N-u(t_i)}^{m_i(t_i)}}{C_N^{m_i(t_i)}} \frac{1}{n_i(t_i)^2 \lambda_2} \tag{17}$$

Because the utility for average delay is different with the one for delivery ratio, the two performances could not be optimized concurrently.

5.4. Parameters Collection

As described in Section 3-A, three parameters $n_i(t_i)$, $m_i(t_i)$, and $u(t_i)$ need to be collected in order to calculate the per-photo utility. Some specialists [43] consider that all the uncertain variables could be achieved by control center. However, actually the design is impractical in the mobile photo crowdsensing. Therefore, we propose a parameter collection way through regarding the PoIs as the common users. Then, all the users collect and exchange their own storage history and mobility history. The communication conditions among the users and PoIs are shown in Figure 9.

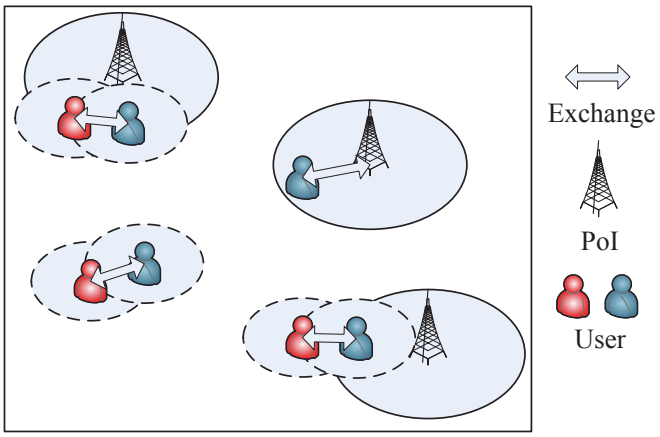
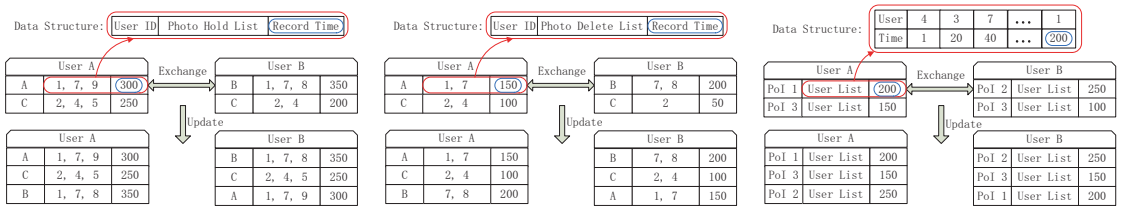


Figure 9. The communication conditions among the users and PoIs. The PoIs and the users are regarded as the same; they collect parameters and exchange with each other.

5.4.1. Collection of $m_i(t_i)$

Every user holds a table as shown in Figure 10a. Obviously, the space occupied by a table is far smaller than the photo size. The $m_i(t_i)$ table includes all photos ever stored, and updating time, which is updated when two tables exchange records; the update process is shown in Figure 10a. It is worth noting that nobody except the initial user could alter the record time; only when a new photo enters its storage could the record then be modified. When two users with the same records encounter each other, they will not exchange photos; however, they could update the newest record time with each. In this way, all the users can estimate $m_i(t_i)$ for photo i.



(**a**) $m_i(t_i)$ exchange (**b**) $d_i(t_i)$ exchange (**c**) $u(t_i)$ exchange

Figure 10. Parameter collection process to estimate $m_i(t_i)$, $d_i(t_i)$, and $u(t_i)$.

5.4.2. Collection of $n_i(t_i)$

According to the definition of $d_i(t_i)$ in Table 1, $m_i(t_i)$ and $n_i(t_i)$ can be associated through Equation (18). Therefore, the problem turns to collect $d_i(t_i)$, which is easier for us to collect compared with $n_i(t_i)$. The collection process of $d_i(t_i)$ is shown in Figure 10b, which is similar to that of $m_i(t_i)$. Every node maintains a data structure including user id, deleted photo list, and record time. The deleted list contains all deleted photos; when users encounter each other, they exchange and update the records. It is worth noting that users will not receive the same photo twice [44]:

$$n_i(t_i) = m_i(t_i) + 1 - d_i(t_i) \tag{18}$$

5.4.3. Collection of $u(t_i)$

Every PoI records the ever-been user list, which includes the user id and the entering time. The data structure is shown in Figure 10c. When a user enters a PoI, the PoI adds the user to the ever-been user list. At the same time, the user takes the records of the PoI and exchanges them with the other users. The exchange process is also shown in Figure 10c. $u(t_i)$ is the number of users that have ever been in any PoI in photo i's elapsed time t_i. After a period of time, every user has the ever-been user list for all the PoIs, so they could calculate the number of users that have ever been in the PoI before the elapsed time t_i. Therefore, $u(t_i)$ is achieved.

Through the above parameter collection methods, $n_i(t_i)$, $m_i(t_i)$, and $u(t_i)$ could be calculated through every user's collected records. Then, every user could achieve the utilities of all the photos stored in its storage. Finally, they could make sending and deleting decisions according to the utility. Thus, a storage management strategy in mobile phones for photo crowdsensing is proposed.

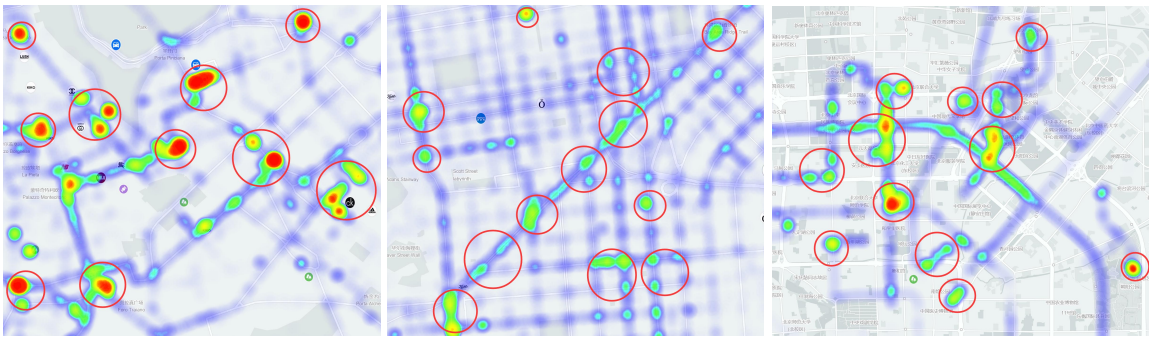
6. Performance Evaluation

6.1. The Traces Used and Settings

The random-waypoint mobility pattern and three real traces, *roma/taxi trace set* [45], *epfl trace set* [46], and *geolife trace set* [47,48], are used to prove the performances of the proposed strategies. Random-waypoint mobility pattern requires each user to keep doing the same actions in a fixed area, walking directly to a certain destination, which is randomly ensured. The *roma/taxi* records 320 users, who work in Rome, Italy. GPS could help record their positions, and the central server could collect all the users' positions. The *epfl* is collected in San Francisco, CA, USA. It is the GPS data from about

500 taxis collected over 30 days in the San Francisco Bay Area. The *geolife* has 17,621 traces, which move about 1.2 million kilometers and continue for about 48,000+ hours.

We filter the above three real-world traces to remove some anomaly trajectories (intermittent or faraway traces). Baidu map is used to carry the GPS traces. Through JavaScript API, we draw the map and corresponding thermodynamic chart (Figure 11), while the PoIs of random-waypoint are set randomly. The simulation parameters in this network environment are shown in Table 2.



(**a**) roma/taxi (**b**) epfl (**c**) geolife

Figure 11. The PoI areas in Baidu map of the three real-world data sets.

Table 2. Simulation Parameters

Parameter	Random-Waypoint	Traces		
		Roma	Epfl	Geolife
Simulation Time	800,850,900,950,1000			
TTL	600∼800	700∼900		400∼600
Time Unit (s)	1	15	30	5
Number of PoIs	10	11	12	12
PoI Radius (m)	300	200	80	300
User Number	100	158	368	727
Connection Range	30	10	10	150
Storage Space	3∼7	5∼9	3∼7	8∼12
Photo Interval	1,2,3,4,5			

6.2. Strategies and Performances in Comparison

To test the proposed storage management strategies, our simulations are divided into the following two parts: (1) performances of the proposed storage management strategies and (2) distribution of utility.

For the first part, we mainly consider the following two proposed management strategies: Storage Management strategies in mobile phones for Photo Crowdsensing to maximize delivery Ratio (SMPC-R) and Storage Management strategies in mobile phones for Photo Crowdsensing to minimize average Delay (SMPC-D). SMPC-R refers to the delivery ratio utility as the priority to send and delete photos, while SMPC-D uses the average delay utility. We attempt to prove that SMPC-R achieves the highest delivery ratio and SMPC-D gets the lowest average delay compared with the other four storage management strategies [29]: (1) DL-Delete Last (or "Delete tail") deletes the just comer photo, (2) DF-Delete Front deletes the photo staying in the buffer for the longest time when the buffer is full, (3) DO-Delete Oldest deletes the photo that has lived for the longest time, and (4) DY-Delete Youngest deletes the photo that is the youngest.

For the second part, we turn our attention to the utility distributions. We want to test whether the distributions calculated in this paper for SMPC-R and SMPC-D are sensitive to different buffer overflown levels. Moreover, we also pay attention to how SMPC performs in the different congestion regimes and whether the resulting utility shape could be approximated by simpler policies.

The following two performances are mainly taken into consideration:

1. Delivery ratio: successfully delivered photo number divided by total number of photos.
2. Average delay: required average time for the photos that have been delivered.

6.3. Simulation Results

6.3.1. Performances of SMPC-R and SMPC-D

To test the utilities of SMPC-R and SMPC-D, we focus on four sets of simulations in terms of the random-waypoint mobility pattern and the three traces: *roma/taxi*, *epfl*, and *geolife*, respectively. SMPC-R refers to the utility of Equation (13) as the priority to decide which photo to send or delete, while SMPC-D considers the utility of Equation (17) as the priority. We consider that the following four factors could influence the network performances: storage space, simulation time, photo TTL, and photo generation interval. Therefore, the performances as a function of the storage space, simulation time, photo TTL, and generation interval are shown in Figures 12–15 for the four data sets, respectively.

As seen in Figure 12, the delivery and delay performances in terms of storage space, simulation time, photo TTL, and generation interval are tested. The ranking of the delivery ratio performances is SMPC-R>DO>DF>SMPC-D>DY>DL, which is reasonable because SMPC-R refers to the utility, which is used to maximize the delivery ratio, as the priority to send or delete photos. In other words, SMPC-R decides whether to send or delete a photo, aiming to maximize the delivery ratio of all the photos. Thus, SMPC-R gets the best delivery performance. Moreover, DO and DF achieve a higher delivery ratio than that of SMPC-D, DY, and DL, which makes sense because DO and DF delete the photos in old age, and the new photos have more opportunities to be stored in storage and delivered to the PoIs.

It is worth noting that delivery ratios grow up with the increase of the storage space for all the storage management strategies. This makes sense and is easy to understand. Moreover, with the growth of simulation time and photo TTL, the delivery ratio also appears as an increasing trend because a longer simulation time or photo TTL leads to a higher probability for delivering the photos. In addition, a higher photo generation interval also means a lower congestion situation, which is similar to that of a larger storage space. Therefore, the delivery ratio also increases with the growth of the photo generation interval for all the storage management strategies.

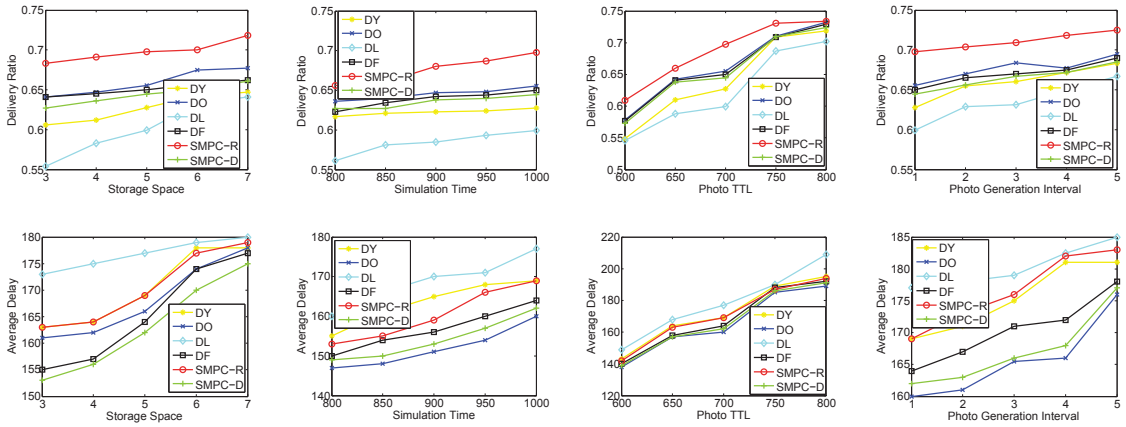


Figure 12. Performance comparisons on the random-waypoint mobility pattern: delivery ratio and average delay.

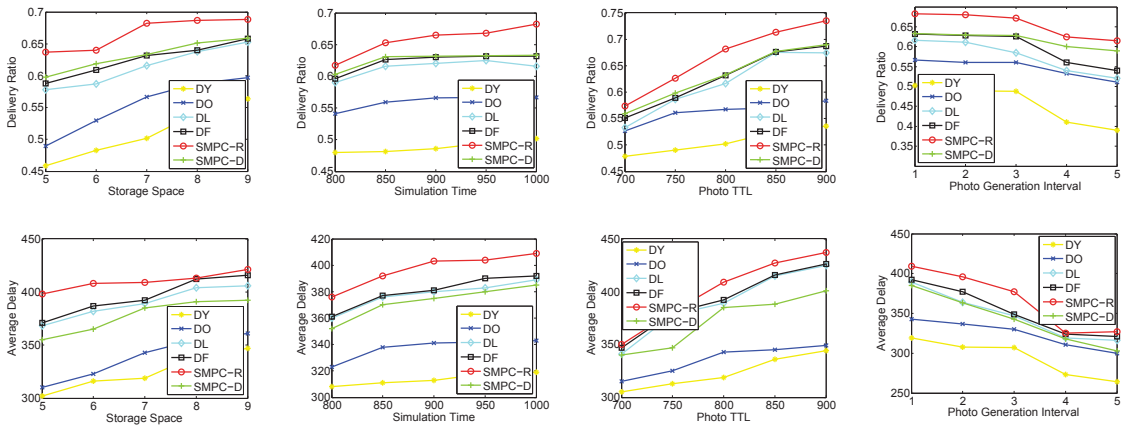


Figure 13. Performance comparisons on the *roma/taxi trace set*: delivery ratio and average delay.

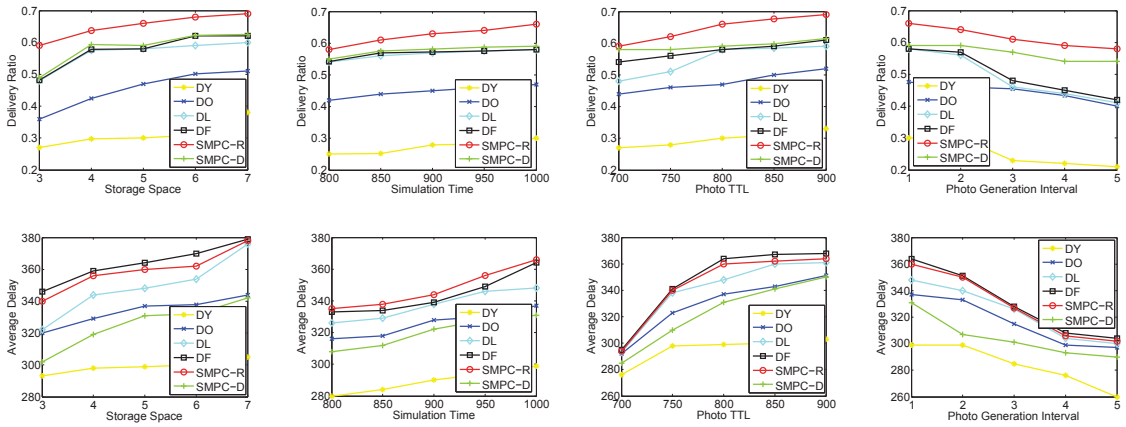


Figure 14. Performance comparisons on the *epfl trace set*: delivery ratio and average delay.

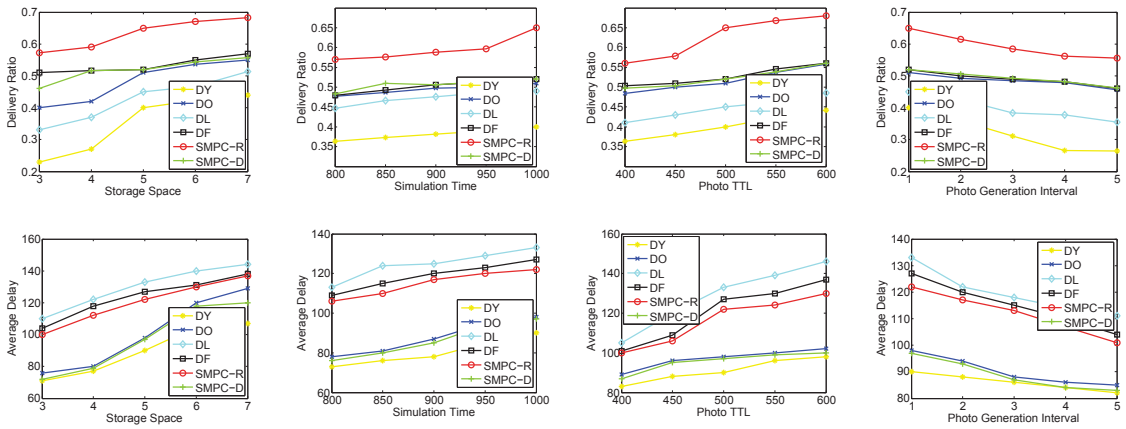


Figure 15. Performance comparisons on the *geolife trace set*: delivery ratio and average delay.

Note that we also test the average delay as a function of storage space, simulation time, photo TTL, and generation interval (as shown in Figure 12). The ranking of delay performances is DL>DY>SMPC-R>DO>DF>SMPC-D, which is reasonable because SMPC-D considers the utility, which is used to minimize the average delay, as the priority to send or delete photos. In other words, SMPC-D decides to send the photo that has the highest influence on decreasing the average delay, and decides to delete the the photo that has the lowest influence on increasing the average delay. Thus, the average delay of SMPC-D is the lowest. Moreover, DL and DY achieve the highest average delay out of all the other storage management strategies because DY and DL delete the new photos, so the average elapsed time of the successfully delivered photos is the highest.

Most importantly, the average delay increases with the growth of the storage space for all the storage management strategies. This is because more storage space leads to a larger space for the photo to be stored, and some photos deleted due to storage overflowing could be delivered when the storage becomes larger. Thus, the average delay of all the successfully delivered photos becomes larger. Moreover, as the simulation time and photo TTL increase, the average delay also appears as an increasing trend because a longer simulation time or photo TTL lead to a higher elapsed time to deliver the photos. In addition, a higher photo generation interval also means a larger storage space, which is explained in the previous paragraph. Therefore, the average delay also increases with the growth of the photo generation interval.

As shown in Figure 13, in the second group simulation, we test the performance comparisons (delivery ratio and average delay) on the on the *roma/taxi trace set* as a function of storage space, simulation time, photo TTL, and generation interval. Obviously, SMPC-R also achieves the highest delivery ratio in terms of storage space, simulation time, photo TTL, and generation interval, while DY also gets the lowest delivery ratio. Actually, DY just deletes the newest photo (the youngest photo) in each mobile phone's storage. Therefore, the delivery ratio of DY is not good in photo crowdsensing. Intuitively, we should protect the new photos (just taken) from being deleted, and delete the old photos when the storage is overflowed. In this way, the photo crowdsensing performance could be better.

Similar results to that of Figure 12 are shown in Figure 13; the average delay of all the storage management strategies also appears in an upward trend along with the increase of the storage space, which is easy to understand. However, SMPC-D achieves a higher average delay compared with DY and DO. Through analysis, in the delivery ratio simulations of *roma/taxi trace set*, DY and DO achieve the lowest results; the definition of average delay is required average time for the photos that have been finished delivering. Thus, the photos that have been delivered have a shorter delivery time, and there are a larger number of undelivered photos. The above reasons explain why SMPC-D does not achieve the lowest average delay. We omit detailed descriptions of Figures 14 and 15 because their simulation results are similar to those of Figures 12 and 13.

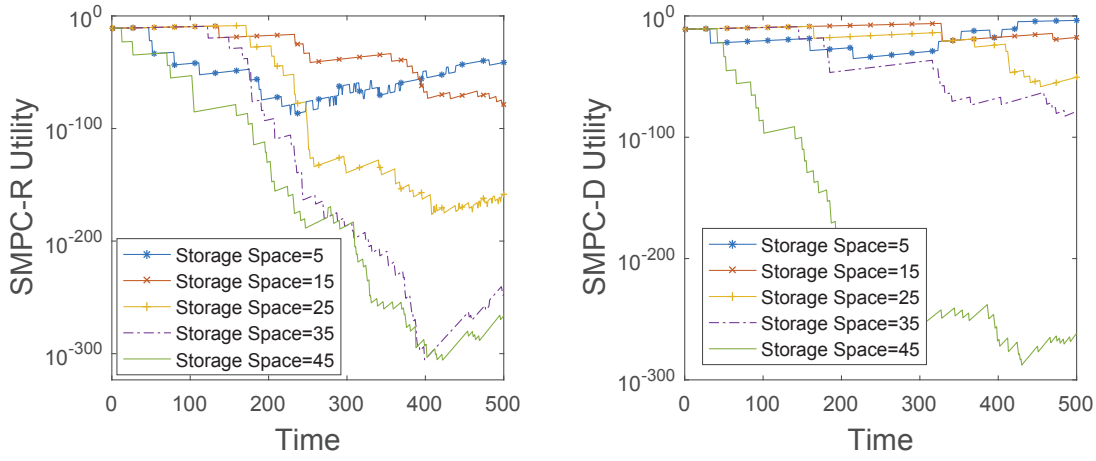
In conclusion, SMPC-R achieves the highest delivery ratio and SMPC-D achieves the lowest average delay.

6.3.2. Distribution of Utility

In this subsection, we focus on the research in terms of the utility distributions. We attempt to show the changes of SMPC-R and SMPC-D utilities along with time t in the different congestion regimes. We also consider whether the resulting utility shape is related to the different congestion levels.

We focus again on the *roma/taxi trace set*. First, we change the storage space from 5 to 45, corresponding to the different congestion regimes. In Figure 16, we plot the utility for maximizing the delivery ratio (SMPC-R) and utility for minimizing the average delay (SMPC-D) described in Sections IV-B and IV-C, as a function of time t. Obviously, the utility distribution has a non-trivial shape, resulting in a complex photo sending and deleting strategy in the mobile phone's storage space.

As shown in Figure 16a, when the storage space is 5, the scenario of a mobile phone is considered as a high congestion. We could see that the SMPC-R utility appears as a complex change along with time t. This precisely helps explain why simple delete and send strategies (e.g., Delete Youngest, Delete Oldest, Delete Front, or Delete Finally, etc.) lead to incorrect decisions during congestion and perform worse than the SMPC-R. When the storage space is 45, the scenario of the mobile phone is considered as a low congestion. We could see that the shape of SMPC-R utility appears as a downward trend along with time t. This suggests that the storage management strategy in low congestion regimes could be approximated by the simpler "Delete Oldest photo" policy, which does not require collecting and exchanging parameters among users.



(**a**) SMPC-R Utility (**b**) SMPC-D Utility

Figure 16. The changes of SMPC-R and SMPC-D utilities along with time t in the different storage spaces.

7. Conclusions

We have looked into the problem of storage management in mobile phones for photo crowdsensing. First, we consider the users with stored photos as the sources, and also consider the PoIs as the destinations. The epidemic exchanging process is implemented to upload the photos from the sources to the destinations. Then, taking the limited storage of mobile phones into consideration, we propose two utility-based storage management strategies in mobile phones for photo crowdsensing: one is for maximizing delivery ratio (SMPC-R) and the other one is for minimizing average delay (SMPC-D). The storage management strategy makes the sending/deleting decisions according to the photo's utility, which is calculated by measuring the impact of copying or deleting the photo on the considered performances. We have done simulations based on the random-waypoint model and three

real traces: *roma/taxi*, *epfl*, and *geolife*. The results show that, compared with other storage management strategies, SMPC-R gets the highest delivery ratio and SMPC-D achieves the lowest average delay.

Author Contributions: Data curation, E.W.; Funding acquisition, Y.Y.; Methodology, Z.Q.; Project administration, Y.Y.; Software, X.M.; Writing: original draft, E.W. and X.L.; Writing: review and editing, X.L., X.M., D.L. and W.M. All authors have read and agreed to the published version of the manuscript.

Funding: There is no founding support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ganti, R.K.; Ye, F.; Lei, H. Mobile crowdsensing: Current state and future challenges. *IEEE Commun. Mag.* **2011**, *49*, 32–39. [CrossRef]
2. Luo, T.; Kanhere, S.S.; Tan, H.P.; Wu, F.; Wu, H. Crowdsourcing with Tullock contests: A new perspective. In Proceedings of the IEEE INFOCOM 2015, Hong Kong, China, 26 April–1 May 2015
3. Zhou, T.; Xiao, B.; Cai, Z.; Xu, M.; Liu, X. From Uncertain Photos to Certain Coverage: a Novel Photo Selection Approach to Mobile Crowdsensing. In Proceedings of the IEEE INFOCOM 2018, Honolulu, HI, USA, 15–19 April 2018
4. Ali, K.; Al-Yaseen, D.; Ejaz, A.; Javed, T.; Hassanein, H.S. CrowdITS: Crowdsourcing in intelligent transportation systems. In Proceedings of the IEEE WCNC 2012, Orlando, FL, USA, 25–30 March 2012.
5. Souliotis, N.; Tsadimas, A.; Nikolaidou, M. Real-time information about public transport's position using crowdsourcing. In Proceedings of the ACM PCI 2014, Athens, Greece, 1–3 October 2014.
6. Liu, C.; Zhang, L.; Liu, Z.; Liu, K.; Li, X.; Liu, Y. Lasagna: Towards Deep Hierarchical Understanding and Searching over Mobile Sensing Data. In Proceedings of the ACM MobiCom 2016, New York, NY, USA, 3 October 2016.
7. Wang, Y.; Hu, W.; Wu, Y.; Cao, G. SmartPhoto: A Resource-Aware Crowdsourcing Approach for Image Sensing with Smartphones. In Proceedings of the ACM MobiHoc 2014, Philadelphia, PA, USA, 11–14 August 2014.
8. Gao, R.; Zhao, M.; Ye, T.; Ye, F.; Wang, Y. Jigsaw: Indoor Floor Plan Reconstruction via Mobile Crowdsensing. In Proceedings of the ACM MobiCom 2014, Philadelphia, PA, USA, 11–14 August 2014.
9. Gong, X.; Chen, X.; Zhang, J.; Poor, H.V. Exploiting Social Trust Assisted Reciprocity (STAR) Toward Utility-Optimal Socially-Aware Crowdsensing. *IEEE Trans. Signal Inf. Process. Over Netw.* **2015**, *1*, 195–208. [CrossRef]
10. Reddy, S.; Estrin, D.; Hansen, M.; Srivastavai, M. Examining Micro-Payments for Participatory Sensing Data Collections. In Proceedings of the ACM UbiComp 2010, Copenhagen, Denmark, 26– 29 September 2010.
11. Zhang, D.; Xiong, H.; Wang, L.; Chen, G. CrowdRecruiter: Selecting Participants for Piggyback Crowdsensing under Probabilistic Coverage Constraint. In Proceedings of the ACM UbiComp 2014, Philadelphia, PA, USA, 11–14 August 2014.
12. Xiong, H.; Zhang, D.; Chen, G.; Wang, L.; Gauthier, V.; Barnes, L.E. iCrowd: Near-Optimal Task Allocation for Piggyback Crowdsensing. *IEEE Trans. Mob. Comput.* **2016**, *15*, 2010–2022. [CrossRef]
13. Guo, B.; Chen, C.; Zhang, D.; Yu, Z. Mobile crowd sensing and computing: when participatory sensing meets participatory social media. *IEEE Commun. Mag.* **2016**, *54*, 131–137. [CrossRef]
14. Singla, A.; Krause, A. Truthful Incentives in Crowdsourcing Tasks using Regret Minimization Mechanisms. In Proceedings of the ACM WWW 2013, Montreal, QC, Canada, 22–24 July 2013.
15. Difallah, D.E.; Catasta, M.; Demartini, G. The Dynamics of Micro-Task Crowdsourcing: The Case of Amazon MTurk. In Proceedings of the ACM WWW 2015, Denver, CO, USA, 12–16 October 2015.
16. Loiseau, P.; Schwartz, G.; Musacchio, J.; Amin, S.; Sastry, S.S. Incentive Mechanisms for Internet Congestion Management: Fixed-Budget Rebate versus Time-of-Day Pricing. *IEEE/ACM Trans. Netw.* **2013**, *22*, 647–661. [CrossRef]
17. Yang, D.; Xue, G.; Fang, G.; Tang, J. Incentive Mechanisms for Crowdsensing: Crowdsourcing With Smartphones. *IEEE/ACM Trans. Netw.* **2015**, pp. 1–13. [CrossRef]
18. Xu, J.; Xiang, J.; Yang, D. Incentive Mechanisms for Time Window Dependent Tasks in Mobile Crowdsensing. *IEEE Trans. Wirel. Commun.* **2015**, *14*, 6353–6364. [CrossRef]

19. Xu, J.; Rao, Z.; Xu, L.; Yang, D.; Li, T. Incentive Mechanism for Multiple Cooperative Tasks with Compatible Users in Mobile Crowd Sensing via Online Communities. *IEEE Trans. Mob. Comput.* **2019**. [CrossRef]
20. Wang, T.; Luo, H.; Zheng, X.; Xie, M. Crowdsourcing Mechanism for Trust Evaluation in CPCS Based on Intelligent Mobile Edge Computing. *ACM TIST* **2019**, *10*, 62:1–62:19. [CrossRef]
21. Zeng, J.; Wang, T.; Lai, Y.; Liang, J.; Chen, H. Data Delivery from WSNs to Cloud Based on a Fog Structure. In Proceedings of the International Conference on Advanced Cloud and Big Data, CBD 2016, Chengdu, China, August 13–16 2016; pp. 104–109.
22. Burleigh, S.; Hooke, A.; Torgerson, L. Delay-tolerant networking: an approach to interplanetary internet. *IEEE Commun. Mag.* **2003**, *41*, 128–136. [CrossRef]
23. Fall, K. A delay-tolerant network architecture for challenged internets. In Proceedings of the ACM SIGCOMM 2003, Miami Beach, FL, USA, 25–29 August 2003; pp. 27–34.
24. Elwhishi, A.; Ho, P.H.; Naik, K.; Shihada, B. A Novel Message Scheduling Framework for Delay Tolerant Networks Routing. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *24*, 871–880. [CrossRef]
25. Wang, E.; Yang, Y.; Wu, J. A Knapsack-based buffer management strategy for delay-tolerant networks. *J. Parallel Distrib. Comput.* **2015**, *86*, 1–15. [CrossRef]
26. Krifa, A.; Barakat, C.; Spyropoulos, T. Optimal Buffer Management Policies for Delay Tolerant Networks. In Proceedings of the IEEE SECON 2008, San Francisco, CA, USA, 16–20 June 2008.
27. Krifa, A.; Barakat, C.; Spyropoulos, T. An Optimal Joint Scheduling and Drop Policy for Delay Tolerant Networks. In Proceedings of the IEEE WOWMOM 2008, Newport Beach, CA, USA, 23–26 June 2008.
28. Krifa, A.; Barakat, C.; Spyropoulos, T. Message Drop and Scheduling in DTNs: Theory and Practice. *IEEE Trans. Mob. Comput.* **2012**, *11*, 1470–1483. [CrossRef]
29. Balasubramanian, A.; Levine, B.N.; Venkataramani, A. DTN Routing as a Resource Allocation Problem. In Proceedings of the ACM SIGCOMM 2007, San Diego, CA, USA, 24–26 October 2007.
30. Matzakos, P.; Spyropoulos, T.; Bonnet, C. Joint Scheduling and Buffer Management Policies for DTN Applications of Different Traffic Classes. *IEEE Trans. Mob. Comput.* **2018**, *17*, 2818–2834. [CrossRef]
31. Zhou, P.; Jiang, S.; Li, M. Urban Traffic Monitoring with the Help of Bus Riders. In Proceedings of the IEEE ICDCS 2015, Columbus, OH, USA, 29 June–2 July 2015.
32. Rula, J.P.; Bustamante, F.E. Crowdsensing Under (Soft) Control. In Proceedings of the IEEE INFOCOM 2015, San Francisco, CA, USA, 10–15 April 2015.
33. Siahaan, E.; Hanjalic, A.; Redi, J. A Reliable Methodology to Collect Ground Truth Data of Image Aesthetic Appeal. *IEEE Trans. Multimed.* **2016**, *18*, 1338–1350. [CrossRef]
34. Wang, X.; Ding, L.; Wang, Q.; Xie, J.; Wang, T.; Tian, X.; Guan, Y.; Wang, X. A Picture is Worth a Thousand Words: Share Your Real-Time View on the Road. *IEEE Trans. Veh. Technol.* **2016**, *66*, 2902–2914. [CrossRef]
35. Wu, Y.; Huang, H.; Wu, N.; Wang, Y.; Bhuiyan, M.Z.A.; Wang, T. An incentive-based protection and recovery strategy for secure big data in social networks. *Inf. Sci.* **2020**, *508*, 79–91. [CrossRef]
36. Wu, Y.; Huang, H.; Wu, Q.; Liu, A.; Wang, T. A risk defense method based on microscopic state prediction with partial information observations in social networks. *J. Parallel Distrib. Comput.* **2019**, *131*, 189–199. [CrossRef]
37. Zhuo, G.; Jia, Q.; Guo, L.; Li, M.; Li, P. Privacy-preserving Verifiable Data Aggregation and Analysis for Cloud-assisted Mobile Crowdsourcing. In Proceedings of the IEEE INFOCOM 2016, San Francisco, CA, USA, 10–15 April 2016.
38. Wu, Y.; Wang, Y.; Hu, W.; Zhang, X.; Cao, G. Resource-Aware Photo Crowdsourcing Through Disruption Tolerant Networks. In Proceedings of the IEEE ICDCS 2016, Nara, Japan, 27–30 June 2016.
39. Wu, H.; Liu, L.; Zhang, X.; Ma, H. Quality of Video Oriented Pricing Incentive for Mobile Video Offloading. In Proceedings of the IEEE INFOCOM 2016, San Francisco, CA, USA, 10–15 April 2016.
40. Zhou, T.; Xiao, B.; Cai, Z.; Xu, M. A Utility Model for Photo Selection in Mobile Crowdsensing. *IEEE Trans. Mob. Comput.* **2019**. [CrossRef]
41. Vahdat, A.; Becker, D. *Epidemic Routing for Partially-Connected Ad Hoc Networks*; Technical Report; Duke University: Duke, UK, 2000.
42. Uddin, M.Y.S.; Ahmadi, H.; Abdelzaher, T. Intercontact Routing for Energy Constrained Disaster Response Networks. *IEEE Trans. Mob. Comput.* **2013**, *12*, 1986–1998. [CrossRef]
43. Yong, L.; Meng, J.Q. Adaptive Optimal Buffer Management Policies for Realistic DTNs. In Proceedings of the IEEE GLOBECOM 2009, Honolulu, HI, USA, 30 November–4 December 2009.

44. Wang, E.; Yang, Y.; Wu, J.; Liu, W. A Buffer Scheduling Method Based on Message Priority in Delay Tolerant Networks. *J. Comput. Sci. Technol.* **2016**, *31*, 1228–1245. [CrossRef]
45. Bracciale, L.; Bonola, M.; Loreti, P.; Bianchi, G.; Amici, R.; Rabuffi, A. CRAWDAD Dataset Roma/Taxi (v. 2014-07-17). Avaliable online: http://crawdad.org/roma/taxi/20140717 (16 March 2020).
46. Piorkowski, M.; Sarafijanovic-Djukic, N.; Grossglauser, M. CRAWDAD Dataset Epfl/Mobility (v. 2009-02-24). Avaliable online: http://crawdad.org/epfl/mobility/20090224 (16 March 2020).
47. Zheng, Y.; Zhang, L.; Xie, X.; Ma., W.Y. Mining interesting locations and travel sequences from GPS trajectories. In Proceedings of the ACM WWW 2009, Sanibel Island, FL, USA, 15–17 July 2009.
48. Zheng, Y.; Li, Q.; Chen, Y.; Xie, X.; Ma., W.Y. Understanding Mobility Based on GPS Data. In Proceedings of the ACM UbiComp 2008, Seoul, Korea, 21–24 September 2008.