OPAT: Optimized Allocation of Time-Dependent Tasks for Mobile Crowdsensing

Yang Huang, Honglong Chen, *Senior Member, IEEE*, Guoqi Ma, Kai Lin, Zhichen Ni, Na Yan, and Zhibo Wang, *Senior Member, IEEE*

***Abstract*—Mobile crowdsensing (MCS) is an emerging paradigm that leverages pervasive smart terminals equipped with various embedded sensors to collect sensory data for wide applications. As the sensing scale increases in MCS, the design of efficient task allocation becomes crucial. However, many prior task allocation schemes, which ignore the time for task-performing, are not applicable to the scenario where mobile users with limited time budgets are able to undertake multiple sensing tasks. In this article, we focus on the task allocation in time dependent crowdsensing systems and formulate the time dependent task allocation problem, in which both the sensing duration and the user's sensing capacity are considered. We prove that the task allocation problem is NP-hard and propose an efficient task allocation algorithm called optimized allocation scheme of time-dependent tasks (OPAT), which can maximize the sensing capacity of each mobile user. The extensive simulations are conducted to demonstrate the effectiveness of the proposed OPAT scheme.**

***Index Terms*—Mobile crowdsensing, task allocation, time budget, time dependent.**

I. Introduction

IN RECENT years, portable smart mobile devices (e.g., smartphones and iPads) become more and more popular and inseparable in daily life. With the rapid development of technology, mobile devices are equipped with numerous and powerful embedded sensors (e.g., camera, microphone, gyroscope, etc.), which can collect a great quantity of information about human activities and surrounding environment cooperatively, making the mobile users carrying with smart mobile devices convert from traditional data consumers to data providers. This promotes mobile crowdsensing (MCS) [1] to emerge as a new paradigm making use of the crowd to collect large-scale and fine-grained sensory data. Unlike traditional static sensor networks, which rely on deploying a large number of sensor nodes in advance, mobile crowdsensing leverages mobile crowd to carry out low-cost, real-time, extensive, and effective sensing activities. Realizing the great potential of the mobile crowdsensing, a wide range of applications have been developed, such as environment monitoring [2], smart city [3], industrial sensing [4], urban sensing [5], and social networks [6], etc.

Typically, there are three roles in mobile crowdsensing systems [7]: task requesters, crowdsensing platform and task participants (i.e., mobile users). Task requesters publish sensing tasks on the platform in the cloud and the platform is responsible for allocating sensing tasks to appropriate task participants. The participants perform the sensing tasks assigned to them, and then upload the sensing data to the platform to earn a certain amount of rewards. Finally, the platform processes the data and submits it to the requesters. Due to the rapid growth of the number of mobile users and sensing tasks in crowdsensing systems, it is of great significance to design proper task allocation mechanisms to match mobile users with suitable sensing tasks.

Recently a lot of efforts [8]–[11] have focused on developing task allocation mechanisms. Generally, the task allocation can be influenced by benefit, cost, energy consumption, time and locations of sensing tasks and mobile users, etc. To this end, some task allocation mechanisms [10], [8] intend to maximize the sensing quality or minimize the incentive cost with different constraints. And there are also some works [12], [13] proposed to reduce the energy consumption. In addition, many researchers focus on the allocation of location-dependent tasks and time-sensitive tasks [14]–[16].

Most of existing works take into account the valid time of sensing tasks, e.g., time-sensitive tasks and delay-tolerant tasks. The time-sensitive tasks are required to be completed immediately due to their emergency. For example, after an earthquake, many relevant tasks can be published, such as collecting the information of damaged infrastructure, reporting the situation of the trapped people and monitoring the traffic condition. Some researchers are committed to designing task allocation schemes for mobile users to guarantee the tasks are completed before their deadlines. In contrast, the delay-tolerant tasks are usually

Manuscript received April 24, 2021; revised June 18, 2021 and June 27, 2021; accepted June 28, 2021. Date of publication July 7, 2021; date of current version December 27, 2021. This work was supported by NSFC under Grant 61772551, Grant 62111530052, Grant U20A20182, and Grant 61872274 and the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-003. Paper no. TII-21-1802. *(Corresponding author: Honglong Chen.)*

Yang Huang, Honglong Chen, Guoqi Ma, Kai Lin, Zhichen Ni, and Na Yan are with the College of Control Science and Engineering, China University of Petroleum (East China), Qingdao 266500, China (e-mail: yanghuang97@outlook.com; honglongchen1984@gmail.com; 2646983760@qq.com; loftyer@163.com; upc-nzc@outlook.com; 17806264216@163.com).

Zhibo Wang is with the Institute of Cyberspace Research, College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: wzb.zju@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2021.3094527.

Digital Object Identifier 10.1109/TII.2021.3094527

for relatively stable objects, such as water pollution monitoring. It is sufficient for the delay-tolerant tasks to be completed within a time period. Some of the delay-tolerant task allocation schemes focus on recommending suitable paths or arranging reasonable schedules for mobile users. However, the above research works only consider the deadline of each task while ignoring the required time duration to perform each task.

We will consider the time dependent crowdsensing systems, in which the sensing tasks are time dependent. In some mobile crowdsensing applications, it is necessary for the mobile users to continuously sense for a sufficient time duration to obtain effective sensing data, for example, traffic monitoring and noise pollution assessment. Such kind of tasks fall into the range of time dependent crowdsensing tasks, the common characteristic of which is that each users working time for each task needs to meet a requirement of specific time duration. In addition, we take into account the time availability of mobile users. Generally, mobile uses are with limited time budgets since they are usually part-time employees to perform the sensing tasks in the spare time to earn extra rewards. As different mobile users may have different time budgets, their sensing capacities are also different.

In this article, we intend to investigate the problem of efficiently allocating sensing tasks to mobile users for the time dependent crowdsensing systems. The main challenges are threefold. First, as different mobile users are equipped with different devices, each sensing task's working time for different mobile users may be different. And the rewards paid to different mobile users for completing a sensing task may also be different as the involved resources or efforts are different. Second, the task allocation scheme should be designed with a global perspective to maximize the platform's profits while satisfying all the constraints, which is NP-hard. Furthermore, multiple mobile users can be coordinated to cooperatively perform the same sensing task, making the associated task allocation problem much more complex.

The main contributions of this article are the following.

1) We investigate and formulate the time dependent task allocation problem for the mobile crowdsensing systems, which is proved to be NP-hard.
2) We refine each sensing task's working time for each mobile user, based on which we characterize the cost of performing a sensing task for each mobile user.
3) We propose an efficient task allocation algorithm called optimized allocation scheme of time-dependent tasks (OPAT), which can make full use of the sensing capacity of each mobile user and maximize the profits of platform.
4) We conduct extensive simulations to validate the effectiveness of the proposed OPAT scheme, the results of which illustrate that the proposed OPAT scheme outperforms the existing one.

The rest of this article is organized as follows. Section II reviews the related work. We present the system overview and formulate the time dependent task allocation problem in Section III. We refine the working time and quantify the cost of each mobile user in Section IV. In Section V, we propose the optimized allocation scheme of time-dependent tasks (OPAT). We conduct the performance evaluation in Section VI. Finally Section VII concludes this article.

II. Related Work

With the development of Internet of Things [17]–[19] and the extensive use of mobile smart devices, mobile crowdsensing technology has received great attention. Meanwhile, new challenges are gradually emerging [7], [11], [14], and [20] among which the efficient task allocation is one of the most significant issues. In recent years, there have been studies on task allocation. Zhang *et al.* [21] proposed a participant selection framework, named CrowdRecruiter, aiming at minimizing incentive payments by selecting participants to satisfy probabilistic coverage constraint. Liu *et al.* [12] linked the potential contribution of users with the energy cost of devices, and aimed to minimize the probability of users rejecting tasks while ensuring the quality of sensing data. Karaliopoulos *et al.* [22] studied the user recruitment for mobile crowdsensing over opportunistic networks, which aimed at minimizing the incentive cost while guaranteeing the full coverage of point-of-interest. However, these works only study the task allocation problem in single task allocation scenario.

Some researchers have studied the problem of multitask allocation and made great contributions. Liu *et al.* [8] studied two biobjective optimization problems on multitask allocation: 1) maximizing the number of accomplished tasks while minimizing the total movement distance and 2) minimizing total incentive cost while minimizing the total traveling distance. For each problem, they proposed two optimized algorithms. Song *et al.* [23] proposed a multitask-oriented participant selection strategy to satisfy the quality-of-information (QoI) requirements of sensing tasks with limited budget when considering different incentive requirements, associated sensing capabilities, and uncontrollable mobility of mobile users. He *et al.* [14] considered location-dependent sensing tasks and the travel budget of each mobile user, and proposed an effective approximation algorithm to maximize the rewards for the MCS platform. Guo *et al.* [24] proposed a worker selection framework, named ActiveCrowd, and studied two multitask allocation situations: 1) for time-sensitive tasks, minimizing the total movement distance to ensure that the task can be completed within the specified time and 2) for delay-tolerant tasks, minimizing the total number of users performing the tasks to reduce the cost. Deng *et al.* [25] studied spatial crowdsourcing problem in which each task is associated with a location and an expiration time. The goal is to find a schedule for workers that maximizes the number of performed tasks.

There are a few works considering the valid time of tasks. Wang *et al.* [26] studied the multiobjective task allocation problem with each task having a valid duration, which aimed to maximize the assigned task coverage and minimize the incentive cost simultaneously. Cheung *et al.* [27] studied the time-sensitive task selection problem and proposed an asynchronous and distributed task selection algorithm. Li *et al.* [15] studied the impact of time constraints in multitask allocation scenario and aimed to maximize the utility of the MCS platform. However, these

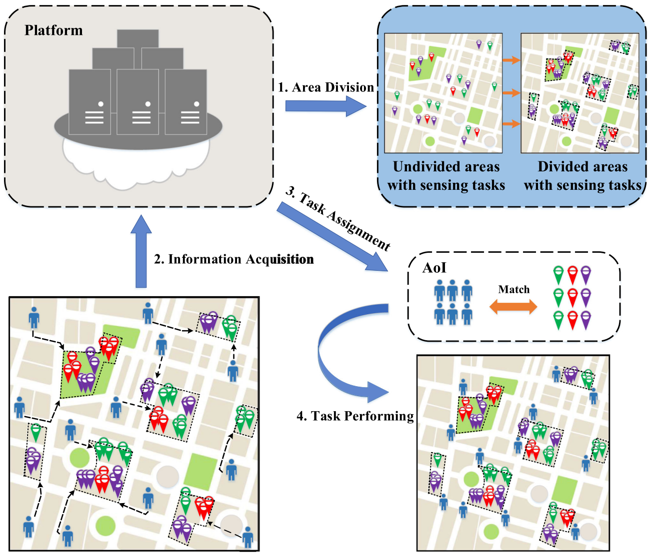


Fig. 1. Four stages in each sensing round—area division stage, information acquisition stage, task assignment stage and task performing stage.

works mainly concerned about the deadline of the task, instead of the duration for completing the task. In fact, few studies take into account the time for mobile users performing sensing tasks, especially when the available time for each mobile user is limited. In this article, we focus on the allocation of time dependent sensing tasks, in which mobile users are with limited sensing capacities. We propose a efficient task allocation scheme that can maximize the profit of platform by making full use of the sensing capacity of each mobile user.

III. System Overview and Problem Formulation

In this section, we first present the system overview of time dependent crowdsensing systems. After that, we formulate the time dependent task allocation problem as an optimization problem.

A. System Overview

We consider a crowdsensing platform which leverages the crowd to collect massive sensing data. The requesters will publish their sensing tasks on the platform and the mobile users can apply for the tasks that they are interested in and get rewards after completing the tasks. The platform is responsible for assigning the tasks to appropriate mobile users.

The operation of the crowdsensing platform is divided into multiple rounds. As shown in Fig. 1, each round is comprised of four stages, including area division, information acquisition, task assignment, and task performing. In the area division stage, each sensing task issued by the requester can be divided into multiple identical subtasks by the platform. After that, the platform divides the large-scale sensing area into multiple subareas, each of which is called as area of interest (*AoI*). In the information acquisition stage, the basic information of new participants will be registered in preparation for task assignment as soon as they join the crowdsensing system. The participants and tasks arriving at the system after the first two stages of the current round will not be processed until task assignment stage of next round. Then, in the task assignment stage, the crowdsensing system performs task allocation in each AoI separately based on the mobile users' information. Finally, each participant works on its allocated tasks and uploads the sensing data in the task performing stage. The unperformed tasks in the current round will be added to the task assignment stage of next round. The above four stages will be repeatedly until all the tasks have been completed or no new participant joins the system.

In the information acquisition stage of each round, each mobile user can upload the necessary information to the platform and select its AoI according to its preference and time budget (e.g., the user is likely to choose the AoI near home or on the way to its destination). Since $AoIs$ are far from each other, the user may not be able to move from one AoI to another within his/her limited time budget, i.e., each user has only one AoI choice. Therefore, we make the following three assumptions.

1) Compared with the working time for sensing tasks, the travel time among the tasks within an AoI can be negligible as the tasks are highly clustered in the area.
2) Each user selects only one AoI according to his/her own preference and upload the time budget for task performing within the AoI.
3) Each user is rational, indicating that it will refuse to perform the tasks if the working time exceeds his/her time budget or the gained reward is lower than its cost. Hence, we will concentrate on the task allocation within each AoI in the following.

B. Problem Formulation

Let $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ denote the set of sensing task in the AoI, where t_j denotes the jth task. Let $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ denote the set of mobile users who select the AoI, where u_i is the ith mobile user. In particular, the sensing tasks are time dependent, which means that each task t_j is associated with a required sensing duration. Specifically, to effectively perform task t_j, user u_i needs to spend a certain amount of working time WT_{ij} (for example, recording a specific-length video). Let $\mathcal{T}_{u_i}$ denote the set of tasks that the platform assigns to user u_i and $WT_{\mathcal{T}_{u_i}}$ denote the total working time user u_i spends in completing $\mathcal{T}_{u_i}$. Each user u_i has a time budget to perform the sensing tasks, denoted as B_{u_i}. According user u_i's rationality, it will perform $\mathcal{T}_{u_i}$ only if $WT_{\mathcal{T}_{u_i}} \leq B_{u_i}$.

Generally, the task requesters require the platform to divide a task into multiple subtasks (i.e., multiple independent measurements) before publishing it to guarantee the sensing quality, and these subtasks have the same measurement requirements and need to be completed by different mobile users. The division of sensing tasks will be carried out in the area division stage. Assume that each task t_j is divided into b_j independent subtasks with identical characteristic, where b_j is associated with the required sensing quality. That is, each user can perform at most one subtask of task t_j and task t_j is required to be performed by b_j users.

Once the tasks are completed, the platform will get revenues from the requesters and pay rewards to the users. Let $P_{ij} = r_{ij}$ - p_{ij} denote the net profit when user u_i performs task t_j, which is the difference between the revenue r_{ij} that the platform gains from the requester and the price p_{ij} that the platform pays to user u_i.

The crowdsensing platform aims to maximize its net profit by assigning tasks to suitable users. To this end, the time dependent tasks allocation problem, named TDTA, can be formulated as follows:

$$TDTA: \ \max \ y(\boldsymbol{\vartheta}) = \sum_{i=1}^{n} \sum_{j=1}^{m} P_{ij} \vartheta_{ij} \tag{1}$$

$$\text{s.t.} \begin{cases} \mathcal{T}_{u_i} = \{t_j \mid \vartheta_{ij} = 1\}, & \forall t_j \in \mathcal{T} \quad \text{(1a)} \\ WT_{\mathcal{T}_{u_i}} \leq B_{u_i}, & \forall u_i \in \mathcal{U} \quad \text{(1b)} \\ \sum_{i=1}^{n} \vartheta_{ij} \leq b_j, & \forall t_j \in \mathcal{T} \quad \text{(1c)} \\ \vartheta_{ij} = \{0, 1\}, & \forall u_i \in \mathcal{U}, t_j \in \mathcal{T} \quad \text{(1d)} \end{cases}$$

where ϑ_{ij} is the decision variable, $\vartheta_{ij} = 1$ indicates task t_j is assigned to user u_i and $\vartheta_{ij} = 0$ otherwise. Constraint (1 b) is to ensure that the total working time required for the assigned tasks will not exceed the user's time budget. In Constraint (1 c), $\sum_{i=1}^{n} \vartheta_{ij} \leq b_j$ since task t_j is divided into b_j subtasks, which require b_j users to perform.

IV. Working Time Refinement and Cost Calculation

In this section, we first refine the working time of each mobile user for performing tasks. Then we characterize the cost of each mobile user by taking important factors into consideration and calculate the reward that the platform should pay to each user.

A. Working Time Refinement

As the sensing tasks published by the platform are with different requirements, they will take different working time for each user. Similarly, for a task, different users require different working time since they are with different capabilities [28], [29].

There are four processes for a user to perform a task, i.e., downloading the task description, sensing, preliminary data preprocessing, and uploading the sensing report. The working time for a user to perform a sensing task is from the beginning of task description file downloading to the end of sensing report submitting. After the platform allocates the tasks, each user will first download the task description file if he/she accepts the proposal. Then it conducts continuous sensing for a specific time duration to perform each allocated task. Furthermore, the tasks may require the users to undertake some data preprocessing, such as video preliminary analysis. Finally, each user uploads its sensing report to the platform. Since the task description file is generally a textual document to explain the task category, task content and task operation requirements, whose size is relatively small, the downloading time can be ignored. Consequently, the total working time that user u_i spends in completing task t_j, consisting of three parts: 1) sensing time; 2) computing time and 3) uploading time, can be formulated as follows:

$$WT_{ij} = ST_{ij} + CT_{ij} + UT_{ij} \tag{2}$$

where ST_{ij}, CT_{ij}, and UT_{ij} represent the sensing time, computing time, and uploading time that user u_i spends on task t_j, respectively.

In area division stage, the platform publishes the time dependent sensing tasks, each of which are modeled as a triplet $t_j = \{D_j, F_j, G_j\}$, where D_j (in bits) denotes the size of sensing data, F_j (in CPU cycles/bit) denotes the number of CPU cycles required to process per bit sensing data, and G_j denotes the processed data (sensing report) size stipulated by the platform.

1) Sensing Time: For each time dependent sensing task, the platform will specify the time for the user to sense to obtain valid sensing data. For example, in traffic monitoring scenario, the platform requires the user to record a specific-duration traffic video in a certain position of a road. The platform can divide this task into multiple subtasks with identical characteristics, i.e., multiple users will be recruited to record the specific-duration traffic videos separately at the same position, and the platform will eventually receive multiple sensing reports from the recruited users. Therefore, we assume that the sensing time ST_{ij} of task t_j is the same for $i = 1, 2, \ldots, n$.

2) Computing Time: The users are required to preprocess the sensing data. The computing time for user u_i to preprocess the sensing data of task t_j can be calculated as

$$CT_{ij} = \frac{D_j F_j}{E_i} \tag{3}$$

where E_i (in CPU cycles/s) is the computation resource of user u_i.

3) Uploading time: The users will upload the sensing reports to the platform after preprocessing the sensing data. The time for user u_i to transmit the sensing report of task t_j can be calculated as

$$UT_{ij} = \frac{G_j}{Q_i} \tag{4}$$

where Q_i is the achievable transmission data rate (in bit/s) from user u_i to the platform, which depends on user u_i's network bandwidth.

B. Cost Calculation

In crowdsensing, the mobile users consume physical resources and make efforts to perform tasks, therefore the users will have no motivation to participate in if the platform does not compensate them. In this section, we quantify the cost of each user by considering several important factors, and then determine the price for the platform to pay each user.

Mobile users will consume nonnegligible resources when performing the sensing tasks, including battery consumption, computing resource, data traffic, time and efforts, etc. Note that the consumed resources of each user for task performing can be divided into two categories, one is related to the mobile devices carried by the user, which is defined as the hardware cost, and the other is related to the efforts, which is defined as the service cost. Therefore, we consider the cost of users as the combination of the hardware cost and service cost. The hardware cost is associated with the level of mobile devices, and the service cost depends on the length of the working time. However, it is difficult to calculate the absolute value of the cost directly. A potential solution is to compare the costs of all the users and then determine the relative level of each cost. Let c_{ij} denote the cost level of user u_i to perform task t_j, then we have

$$c_{ij} = \alpha H_i + \beta S_{ij} \tag{5}$$

where H_i represents the hardware cost level of user u_i and S_{ij} represents the service cost level of user u_i to perform task t_j. α and β are the weights satisfying $\alpha + \beta = 1$, the values of which are determined by the platform. In this article, we simply consider that the hardware cost level and service cost level have equivalent impact on the cost level, thus $\alpha = \beta = 0.5$.

Next, we calculate the hardware cost level by considering three different parameters associated with mobile devices: number of sensors, computation resource, and transmission data rate. In general, the number of sensors of a mobile device will affect the hardware cost level, since a mobile device equipped with multiple sensors can provide high-quality sensing data with a high cost; the computation resource will indirectly impact the battery energy available with mobile users and the cooling systems of mobile devices; the transmission data rate is associated with the network bandwidth of the user, which is related to the data traffic costs. Therefore, we use a linear function to numerically formulate the impacts of these three parameters on the hardware cost level as follows:

$$H_i = w_1 \frac{N_i}{N_{\max}} + w_2 \frac{E_i}{E_{\max}} + w_3 \frac{Q_i}{Q_{\max}} \tag{6}$$

where N_i is the number of sensors equipped on the mobile device carried by user u_i, and $N_{\max}$ is the maximum equipped number of sensors of all the users. $E_{\max}$ and $Q_{\max}$ are the maximum values of computation resource and transmission data rate of all the users, respectively. w_1, w_2 and w_3 are the weights to measure the relative importance of the three parameters and $w_1 + w_2 + w_3 = 1$.

Similarly, we use the relative length of working time to represent the service cost level for user u_i to performs task t_j as follows:

$$S_{ij} = \frac{WT_{ij}}{WT_j^{\max}} \tag{7}$$

where $WT_j^{\max}$ is the maximum working time spent on task t_j of all the users.

We adopt the analytic hierarchy process (AHP) [30] to calculate the weights of the three parameters. We use X_1, X_2, and X_3 to represent the number of sensors, computation resource and transmission data rate respectively, and $\mathrm{W} = (w_1, w_2, w_3)^T$ represents the vector of weights for the three parameters. Pairwise comparison matrix [31] can be used to evaluate the relative importance among the parameters. The pairwise comparison matrix $A = (a_{ij})_{3\times 3}$ in Fig. 2(a) is an intuitive example, e.g., $a_{13} = 3$ indicates X_1 (i.e., the number of sensors) is slightly more important than X_3 (i.e., the transmission data rate).

Then, we normalize each column of pairwise comparison matrix A, i.e., each element is calculated as $\overline{a_{ij}} = \frac{a_{ij}}{\sum_{k=1}^{3} a_{kj}}$. The normalized pairwise comparison matrix is shown in Fig. 2(b).

TABLE I
SIMULATION PARAMETERS

Parameters	Values
D_j	[50, 100] Mbits
G_j	[10, 20] Mbits
F_j	[200, 300] CPU cycles/bit
ST_{ij}	[1, 3] minutes
E_i	[200, 400] MHz
Q_i	[0.1, 0.5] Mbit/s
N_i	[1, 10]

PAIRWISE COMPARISON MATRIX

	X_1	X_2	X_3
X_1	1	2	3
X_2	1/2	1	2
X_3	1/3	1/2	1

(a)

NORMALIZED PAIRWISE COMPARISON MATRIX

	X_1	X_2	X_3
X_1	0.55	0.57	0.50
X_2	0.27	0.29	0.33
X_3	0.18	0.14	0.17

(b)

Fig. 2. Pairwise comparison matrix and normalized pairwise comparison matrix.

And the vector of weights $\mathrm{W} = (w_1, w_2, w_3)^T$ can be calculated by averaging the elements on each row of the normalized pairwise comparison matrix, i.e.,

$$w_i = \frac{1}{3} \sum_{j=1}^{3} \overline{a_{ij}}. \tag{8}$$

We can get the vector of weights $W = (0.54, 0.30, 0.16)^T$ for the three parameters in Fig. 2(b). With the obtained W, the hardware cost level can be calculated based on (6), after which the cost level can be obtained according to (5). Then, we discuss the following rule to determine the real cost based on the cost level as:

$$C_{ij} = c_0 + \epsilon c_{ij} \tag{9}$$

where C_{ij} is the real cost for user u_i to perform task t_j, c_0 is the basic cost and ϵ is a coefficient linking the cost level of the user with the real cost, and the value of ϵ is determined by the platform.

In addition, we discuss the determination of the price that the platform pays to each user. For simplicity, we make the following assumptions.

1) Since each user chooses the AoI according to its interest and time budget, it will be interested in performing every task in the AoI if he/she has sufficient time budget.
2) The users do not collude with each other.
3) The platform bargains with each user to determine the price to pay and the agreement on one user will not impact that on others.

We adopt the Nash bargaining solution in [14] to calculate the price paid to each user as follows:

$$p_{ij} = \frac{r_{ij} + C_{ij} - \sqrt{\frac{n-1}{n+1}}(r_{ij} - C_{ij})}{2} \tag{10}$$

where r_{ij} is the revenue that the platform will gain from the requester for allocating task t_j to user u_i, C_{ij} is the cost for user u_i to perform task t_j, and n is the number of users selecting the AoI.

V. Task Allocation

In this section, we prove that the TDTA problem is NP-hard and propose the optimized allocation scheme of time-dependent tasks (OPAT).

Theorem 1: The TDTA problem is NP-hard.

Proof: We prove Theorem 1 by degenerating TDTA to a knapsack problem. Please see Appendix A for a proof. ■

In the following, we propose an efficient task allocation algorithm called optimized allocation scheme of time-dependent tasks (OPAT), which can take full advantage of the sensing capacity of each mobile user. We provide the detailed description of the proposed OPAT scheme in the following three steps, which are shown in Algorithm 1.

Algorithm 1: OPtimized Allocation Scheme of Time-Dependent Tasks (OPAT).

Input: $u_i \in \mathcal{U}, t_j \in \mathcal{T}, B_{u_i}, b_j, r_{ij}, p_{ij}, D_j, F_j, Q_j, E_i, G_j$.
Output: S^F.

Step 1: Transform the TDTA problem into the C-TDTA problem;
Step 2: Obtain the preliminary task allocation by solving the knapsack problem of each mobile user;
Initially, $h = 0$;
repeat
  $h = h + 1$;
  Obtain $\mathcal{T}_h$ by selecting m subtasks from M tasks according to (13);
  Solve the knapsack problem associated with mobile user u_h and get TS_h;
  Define $q_h(\mathbf{v})$ and z_{ijk}^h according to (14) and (15);
  Obtain $q_{h'}(\mathbf{v})$ and $z_{ijk}^{h'}$ for next iteration;
until $h = n$ or $q_{h'}(\mathbf{v})$=0.
Step 3: Conflict elimination and task reallocation;
Strategy I:
for $h = 1$ to n **do**
  Redetermine $\mathcal{T}_h$ by removing and adding;
  Solve the knapsack problem of user u_h with $\mathcal{T}_h^I$;
  Update TS_h;
  $h = h + 1$;
end for
The output of Strategy I: S^I
Strategy II:
for $h = n$ to 1**do**
  Redetermine $\mathcal{T}_h$ by removing and adding;
  Solve the knapsack problem of user u_h with $\mathcal{T}_h^{II}$;
  Update TS_h;
  $h = h - 1$;
end for
The output of Strategy II: S^{II}
Compare the profits of S^I and S^{II} and return the allocation with more profits.

Step 1: We reformulate the TDTA problem. To reduce the error and guarantee the quality of sensing, each task t_j is divided into b_j subtasks with identical characteristic. Let t_{jk} denote the kth subtask of the jth task, where $j = 1, 2, \ldots, m$, and $k = 1, 2, \ldots, b_j$. Thus, The total number of subtasks in the system is $M = \sum_{j=1}^{m} b_j$. We assume that the platform will gain net profit z_{ijk} when user u_i completes the subtask t_{jk}, and $z_{ijk} = P_{ij}$ for all $k = 1, 2, \ldots, b_j$. Note that each user can only perform one of the subtasks of each task. Hence, a task allocation scheme is necessary to maximize the platform's benefits. Consequently, we can get a constrained TDTA problem named C-TDTA that is more complicated than the original TDTA problem, which can be formulated as follows:

$$C - TDTA: \ \max\ q(\mathbf{v}) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{b_j} z_{ijk} v_{ijk} \tag{11}$$

$$\text{s.t.} \begin{cases} \mathcal{T}'_{u_i} = \{t_{jk} \mid v_{ijk} = 1, j = 1, 2, \ldots, m, \\ \qquad\qquad\qquad\qquad k = 1, 2, \ldots, b_j\} & \text{(11a)} \\ WT_{\mathcal{T}'_{u_i}} \leq B_{u_i},\ \forall u_i \in \mathcal{U} & \text{(11b)} \\ \sum_{i=1}^{n}\sum_{k=1}^{b_j} v_{ijk} \leq b_j,\ j = 1, 2, \ldots, m & \text{(11c)} \\ \sum_{i=1}^{n} v_{ijk} \leq 1,\ j = 1, 2, \ldots, m,\ k = 1, 2, \ldots, b_j & \text{(11d)} \\ \sum_{k=1}^{b_j} v_{ijk} \leq 1,\ i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, m & \text{(11e)} \\ v_{ijk} = \{0, 1\} & \text{(11f)} \end{cases}$$

where v_{ijk} is the decision variable, $v_{ijk} = 1$ represents that the platform assigns task t_{jk} to user u_i and $v_{ijk} = 0$ otherwise. Constraint (11d) is to ensure that subtask t_{jk} can only be assigned to at most one user since the subtask cannot be divided and completed repeatedly. Constraint (11 e) requires that user u_i can only complete one of the subtasks of task t_j, since the data validity can be guaranteed by multiple sensing data from different users.

Step 2: The preliminary task allocation is obtained by solving the knapsack problem of each mobile user. Next, we introduce how to solve the knapsack problem of each user through iterations. Let $q_{h'}(\mathbf{v})$ denote the reward function at the beginning of iteration h and $z_{ijk}^{h'}$ is the profit associate with $q_{h'}(\mathbf{v})$, i.e., at the beginning of iteration one, the reward function is

$$q_{1'}(\mathbf{v}) = q(\mathbf{v}) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{b_j} z_{ijk} v_{ijk} = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{b_j} z_{ijk}^{1'} v_{ijk}. \tag{12}$$

In each iteration, the reward function will be modified. Let $q_h(\mathbf{v})$ denote the modified reward function at iteration h and z_{ijk}^h is the profit associate with $q_h(\mathbf{v})$. In each iteration h, m subtasks are selected from the total M tasks, in which at most one subtask can be selected from its associated task. The subtask selection rule is: for each task t_j, choose subtask $t_{jk'}$, where

$$k' = \arg \max_{k=1,2,\ldots,b_j} z_{hjk}^{h'}. \tag{13}$$

Let $\mathcal{T}_h$ represent the set of tasks selected in iteration h. After that is to solve the knapsack problem of user u_h. Intuitively, the sensing tasks are items and the time budget of u_h is a knapsack. The platform needs to assign the sensing tasks in $\mathcal{T}_h$ to user u_h to maximize profits without exceeding the capacity of the knapsack. The task set assigned to user u_h, denoted as TS_h, can be obtained by an algorithm of knapsack problem.

Next, we define a new reward function $q_h(\mathbf{v})$ at iteration h as

$$q_h(\mathbf{v}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{b_j} z_{ijk}^{h} v_{ijk} \tag{14}$$

where

$$z_{ijk}^{h} = \begin{cases} z_{hjk}^{h'}, & i = h \vee (i > h \wedge t_{jk} \in TS_h) \\ 0, & \text{otherwise} \end{cases}. \tag{15}$$

Then we can get the reward function at the beginning of next iteration, i.e., $q_{(h+1)'}(\mathbf{v}) = q_{h'}(\mathbf{v}) - q_h(\mathbf{v})$. Accordingly, $z_{ijk}^{(h+1)'} = z_{ijk}^{h'} - z_{ijk}^{h}$. The iterative process terminates when $h = n$ or $q_{h'}(\mathbf{v}) = 0$, which indicates the knapsack problem of each user is solved and a preliminary task allocation $TS = \{TS_1, TS_2, \ldots, TS_n\}$ is obtained.

Step 3: Conflict elimination and task reallocation. Since subtask t_{jk} may be assigned to multiple users in Step 2, which is not allowed, we call it a conflict. Then we have to eliminate the conflicts together with the task reallocation. We put forward the following two conflict elimination and task reallocation strategies.

Strategy I: Eliminating conflicts and reallocating tasks forwards. For user u_h, we reselect the tasks from the total M tasks. First, we redetermine the selected task set by removing some tasks from $\mathcal{T}_h$. The removed task t_{jk} satisfies $t_{jk} \in \{TS_1, TS_2, \ldots, TS_{h-1}, TS_{h+1}, TS_{h+2}, \ldots, TS_n\}$, with which there are two scenarios, one is that task t_{jk} is assigned to u_h and other users, and the other is that task t_{jk} is assigned to other users except u_h. Then, we add some tasks to $\mathcal{T}_h$. The added task t'_{jk} satisfies $t'_{jk} \notin \{TS_1, TS_2, \ldots, TS_n\}$, which means task t'_{jk} is not assigned to any user in Step 2. Note that the added sensing tasks may be different subtasks of the same task. In view of this, we choose task t'_{jk} with a smaller k, e.g., if task t'_{62} and task t'_{63} are not selected in Step 2, only task t'_{62} will be added to $\mathcal{T}_h$. After the above operation, we get a new selected task set $\mathcal{T}_h^I$. Then, we solve the knapsack problem of user u_h with $\mathcal{T}_h^I$ and update the task set TS_h assigned to user u_h. *Strategy I* eliminates the conflicts and reallocates sensing tasks to users from u_1 to u_n. Finally, we can get the task allocation with *Strategy I*, denoted as $S^I = \{S_1^I, S_2^I, \ldots, S_n^I\}$.

Strategy II: Eliminating conflicts and reallocating tasks backwards. All the operations of *Strategy II* are the same as that of *Strategy I*, except we eliminate conflicts and reallocate sensing tasks to users from u_n to u_1, which is the reverse of the assignment order in *Strategy I*. Therefore, we skip the detailed descriptions of *Strategy II*. Also, we can get the task allocation with *Strategy II*, denoted as $S^{II} = \{S_1^{II}, S_2^{II}, \ldots, S_n^{II}\}$.

At the end of Step 3, the profits of S^I and S^{II} will be compared to get the allocation with more profits, denoted as $S^F = \{S_1^F, S_2^F, \ldots, S_n^F\}$.

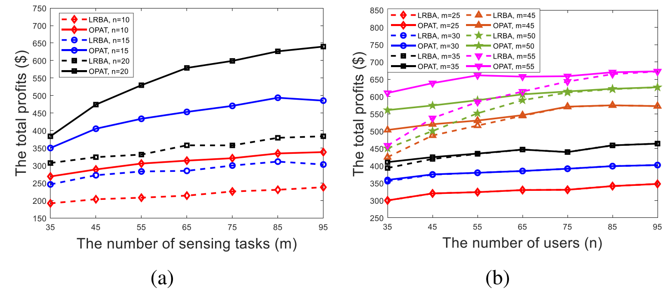


Fig. 3. Comparison of OPAT and LRBA on the total profits with different numbers of tasks and users.

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed OPAT scheme. We first describe the simulation settings. Afterward we compare OPAT with local ratio-based algorithm (LRBA) proposed in [14] via extensive simulations to demonstrate its advantages.

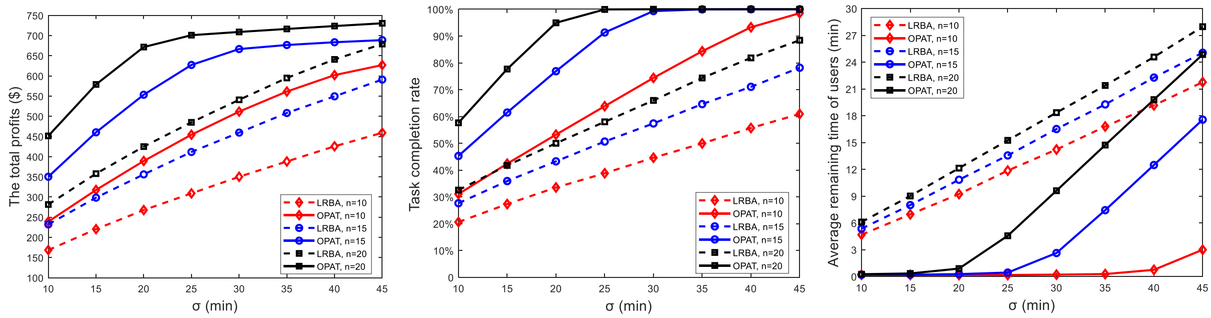
A. Simulation Settings

For each sensing task t_j, the number of subtasks b_j is randomly set within the range of $[1,\ 3]$. The revenue r_{ij} the platform gains from the requester is randomly generated within the range of $[11,\ 16]$. We set $c_0 = 0.5$ and $\epsilon = 10$. The time budget B_{u_i} of user u_i is set as a random number $\sigma + \delta$, where σ is a constant and δ is randomly generated between $[0,\ 5]$. The setting of other parameters are shown in Table I. The uniform distribution is adopted in experimental data and each simulation result is obtained by averaging 50 independent runs.

B. Simulation Results

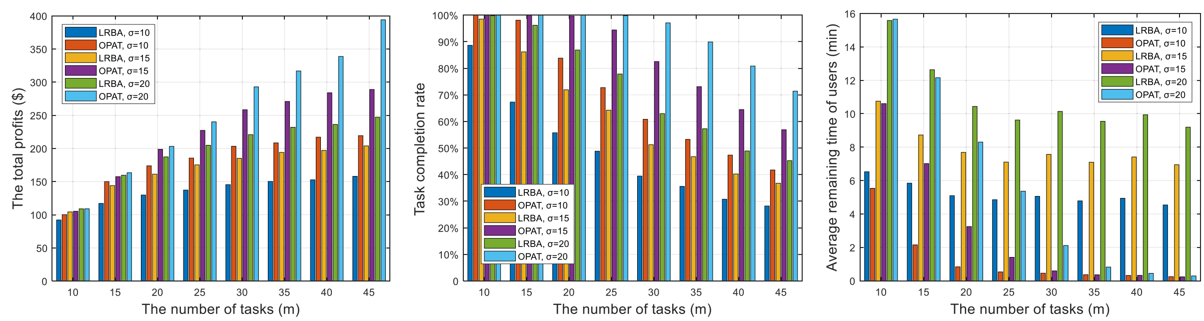
Fig. 3(a) compares the total profits of the OPAT and LRBA schemes when the number of tasks varies. The number of tasks varies from 35 to 95 and the number of users is set as 10, 15, and 20, respectively. σ is set as 15. Fig. 3(a) illustrates that OPAT always achieves more profits than LRBA and the total profits of both the OPAT and LRBA increase with the number of tasks. As OPAT can not only eliminate the conflicts, but also reallocate the previously unassigned tasks to mobile users, it can obtain more profits than LRBA.

Fig. 3(b) shows the comparison of the total profits of the OPAT and LRBA with different numbers of users. The number of users varies from 35 to 95 and the number of tasks is set as 25, 30, 35, 45, 50, and 55, respectively. Fig. 3(b) shows an upward trend when the number of users increases, the reason behind which is that with more tasks, both the OPAT and LRBA can recruit more users and assign the tasks to them to obtain more profits. Moreover, OPAT can achieve more profits than LRBA. When the number of tasks is larger than 45, the performance of OPAT is much better than that of LRBA. This is because when the number of tasks becomes larger, there will be more conflicts



(a) (b) (c)

Fig. 4. Comparison of OPAT and LRBA on the total profits, task completion rate and average remaining time of users with different time budgets.



(a) (b) (c)

Fig. 5. Comparison of OPAT and LRBA on the total profits, task completion rate and average remaining time of users with different numbers of tasks.

during the allocation and more tasks will be dropped, in which OPTA demonstrates its advantages by reallocating the dropped tasks to users.

To comprehensively evaluate the performance, we introduce the following two metrics.

1) Task completion rate: The ratio of the number of completed tasks to the total number of tasks M.
2) Average remaining time of users: The average difference between the time budgets of users and the time spent in completing the assigned tasks.

Fig. 4 shows the impacts of time budget on the performance of the total profits, task completion rate and average remaining time of users. We fix the number of tasks as $m = 65$, then vary σ from 10 to 45 and set the number of users n to 10, 15 and 20, respectively. Fig. 4(a) illustrates that OPAT always obtains more profits than LRBA. And the total profits increase with the time budget. Fig. 4(b) shows that the task completion rate of OPAT is always higher than that of LRBA, which is consistent with the result that OPAT achieves higher profits. Note that when n is relatively large (i.e., $n \geq 15$), the task completion rate of OPAT can reach 100%, which indicates that all of the tasks are successfully allocated to approximately maximize the profits. Fig. 4(c) illustrates that the average remaining time of users of OPAT is less than that of LRBA as OPAT can make full use of the sensing capacities of users via reallocation. With the increase of time budgets of users, the average remaining time of users increases quickly, since all of the tasks have already been allocated and each user does not need to use up his/her time budget.

Fig. 5 shows the impacts of the number of tasks on the performance of the total profits, task completion rate and average remaining time of users. We fix the number of users as $n = 10$, then vary the number of tasks m from 10 to 45 and set σ to 10, 15 and 20, respectively. Fig. 5 illustrates that the task completion rates of OPAT and LRBA approximate 100% when the number of tasks m is relatively small, in which the OPAT and LRBA obtain the similar profits and each user has relatively much remaining time since he/she does not need to use up his/her time budget. The profits of the OPAT and LRBA increase with the number of sensing tasks, while the former increases faster since the OPAT can make full use of the time budget of each user, especially when there are sufficient tasks. However, LRBA fails to fully utilize the users' sensing capacities, which can be validated in Fig. 5(b) and (c). The task completion rate of LRBA decreases significantly with the increase of the number of sensing tasks, while the average remaining time of users decreases more slowly. The reason behind is that when the number of sensing tasks increases, there are more conflicts in task allocation and LRBA cannot successfully allocate all the tasks to the users.

Next, we investigate the differences among the individual profit of each mobile user obtained with the OPAT and LRBA schemes, i.e., to evaluate the fairness of the OPAT and LRBA. To this end, we adopt the following two metrics.

1) Relative standard deviation (RSD)

$$RSD = \frac{\sqrt{\frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{n-1}}}{\bar{x}} \times 100\% \quad (16)$$

2) Relative variance ratio (RVR)

$$RVR = \frac{\sum_{i=1}^{n}|x_i-\bar{x}|}{n\bar{x}} \times 100\% \quad (17)$$

where x_i is the individual profit each mobile user u_i obtains, and $\bar{x}$ is the average profit over all of the n mobile users.

The statistics of the RSD and RVR of the LRBA and OPAT with the same settings in Fig. 5 are conducted and the results are shown in Fig. 2(a) and III. It is illustrated that the RSD and RVR of LRBA are relatively large with each parameter setting, indicating that LRBA does not have the property of fair allocation. The reason behind is that LRBA always assigns a task to the user with a higher profit. The RSD and RVR of OPAT are smaller than that of LRBA with each parameter setting, since OPAT can not only eliminate conflicts, but also reallocate the sensing tasks. For example, if task t_1 is assigned to two users u_1 and u_2 and the profit of assigning task t_1 to u_1 is higher than that to u_2, LRBA will always assign t_1 to u_1, making u_2 lose the opportunity to perform t_1. However, under such scenario OPAT can allocate other sensing tasks to u_2 via reallocation, making u_2 be able to make full use of his/her time budget to obtain more profits. Therefore, we can claim that OPAT outperforms LRBA on the fairness.

VII. Conclusion

In this article, we focused on the problem of allocating time dependent sensing tasks in crowdsensing systems. We first formulated the time dependent task allocation problem and proved it is NP-hard. We then proposed an efficient task allocation algorithm called OPAT, which can make full use of the sensing capacity of each mobile user and maximize the platform's profits. We conducted extensive simulations to evaluate the performance of the proposed OPAT scheme and the simulation results illustrate the effectiveness of OPAT.

Our future work will focus on the higher-level user-AoI task allocation to schedule the user-task match among $AoIs$ based on spatio-temporal characteristics.

Appendix

A. Proof of Theorem 1

Proof: We consider a special case where there is only one user u_1 in the AoI. In this case, the TDTA problem in (1) can be described as follows. The platform publishes a set $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ of m sensing tasks. For $t_j \in \mathcal{T}$, when user u_1 performs task t_j, the working time WT_{1j} needs to satisfy the requirement. User u_1 has a time budget B_{u_1} to perform tasks, and the platform will obtain net profit P_{1j} once task t_j is completed by user u_1. The goal of the platform is to maximize its net profit by selecting a subset of $\mathcal{T}$ for user u_1 with the prerequisite that u_1's working time does not exceed B_{u_1}. Then the TDTA problem can be formulated as follows:

$$\max \quad y(\boldsymbol{\vartheta}) = \sum_{j=1}^{n} P_{1j}\vartheta_{1j}$$

$$\text{s.t.} \begin{cases} \sum_{j=1}^{m} WT_{1j}\vartheta_{1j} \le B_{u_1} \\ \vartheta_{1j} = \{0,1\}, j = 1, \ldots, m \end{cases}.$$

According to [32], the special case of the TDTA problem is a knapsack problem, which is NP-hard. Hence, we can claim that the TDTA problem in (1) is also NP-hard. ■

References

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, Nov. 2011.
[2] P. Dutta *et al.*, "Common sense: Participatory urban sensing using a network of handheld air quality monitors," in *Proc. ACM SenSys*, 2009, pp. 349–350.
[3] X. Kong, J. Cao, H. Wu, and C. H. Hsu, "Mobile crowdsourcing and pervasive computing for smart cities," *Pervasive Mobile Comput.*, vol. 61, pp. 101–114, 2020.
[4] E. Wang *et al.*, "Deep learning-enabled sparse industrial crowdsensing and prediction," *IEEE Trans. Ind. Informat.*, vol. 17, no. 9, pp. 6170–6181, Sep. 2021.
[5] F. Xia, A. Rahim, X. Kong, M. Wang, Y. Cai, and J. Wang, "Modeling and analysis of large-scale urban mobility for green transportation," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1469–1481, Apr. 2018.
[6] F. Xia, L. Liu, B. Jedari, and S. K. Das, "PIS: A multi-dimensional routing protocol for socially-aware networking," *IEEE Trans. Mobile Comput.*, vol. 15, no. 11, pp. 2825–2836, Nov. 2016.
[7] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. ACM 18th Annu. Int. Conf. Mobile Comput. Netw.*, 2012, pp. 173–184.
[8] Y. Liu *et al.*, "TaskMe: Multi-task allocation in mobile crowd sensing," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2016, pp. 403–414.
[9] L. Wang, D. Yang, X. Han, D. Zhang, and X. Ma, "Mobile crowdsourcing task allocation with differential-and-distortion geo-obfuscation," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 2, pp. 967–981, Mar./Apr. 2021.
[10] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," in *Proc. IEEE 33rd Int. Conf. Data Eng.*, 2017, pp. 997–1008.
[11] Q. Tao, Y. Tong, Z. Zhou, Y. Shi, L. Chen, and K. Xu, "Differentially private online task assignment in spatial crowdsourcing: A tree-based approach," in *Proc. IEEE Int. Conf. Data Eng.*, 2020, pp. 517–528.
[12] C. H. Liu, B. Zhang, X. Su, J. Ma, W. Wang, and K. K. Leung, "Energy-aware participant selection for smartphone-enabled mobile crowd sensing," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1435–1446, Sep. 2017.
[13] N. D. Lane *et al.*, "Piggyback crowdsensing (PCS): Energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities," in *Proc. 11th ACM Conf. Embedded Netw. Sensor Syst.*, 2013, pp. 1–14.
[14] S. He, D. Shin, J. Zhang, and J. Chen, "Toward optimal allocation of location dependent tasks in crowdsensing," in *Proc. IEEE Conf. Comput. Commun.*, 2014, pp. 745–753.
[15] X. Li and X. Zhang, "Multi-task allocation under time constraints in mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1494–1510, Apr. 2021.
[16] R. Estrada, R. Mizouni, H. Otrok, A. Ouali, and J. Bentahar, "A crowd-sensing framework for allocation of time-constrained and location-based tasks," *IEEE Trans. Serv. Comput.*, vol. 13, no. 5, pp. 769–785, Sep./Oct. 2020.
[17] X. Ai, H. Chen, K. Lin, Z. Wang, and J. Yu, "Nowhere to hide: Efficiently identifying probabilistic cloning attacks in large-scale RFID systems," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, no. 7, pp. 714–727, Jul. 2021.
[18] H. Chen *et al.*, "DAP: Efficient detection against probabilistic cloning attacks in anonymous RFID systems," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2021.3072929.

[19] X. Kong *et al.*, "Mobile edge cooperation optimization for wearable Internet of Things: A network representation-based framework," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5050–5058, Jul. 2021.
[20] E. Wang, Y. Yang, J. Wu, W. Liu, and X. Wang, "An efficient prediction-based user recruitment for mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 1, pp. 16–28, Jan. 2018.
[21] D. Zhang, H. Xiong, L. Wang, and G. Chen, "CrowdRecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2014, pp. 703–714.
[22] M. Karaliopoulos, O. Telelis, and I. Koutsopoulos, "User recruitment for mobile crowdsensing over opportunistic networks," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 2254–2262.
[23] Z. Song, C. H. Liu, J. Wu, J. Ma, and W. Wang, "QOI-aware multitask-oriented dynamic participant selection with budget constraints," *IEEE Trans. Veh. Technol.*, vol. 63, no. 9, pp. 4618–4632, Nov. 2014.
[24] B. Guo, Y. Liu, W. Wu, Z. Yu, and Q. Han, "Activecrowd: A framework for optimized multitask allocation in mobile crowdsensing systems," *IEEE Trans. Hum.-Mach. Syst.*, vol. 47, no. 3, pp. 392–403, Jun. 2017.
[25] D. Deng, C. Shahabi, and U. Demiryurek, "Maximizing the number of workers self-selected tasks in spatial crowdsourcing," in *Proc. ACM Sigspatial Int. Conf. Adv. Geographic Inf. Syst.*, 2013, pp. 324–333.
[26] L. Wang, Z. Yu, Q. Han, B. Guo, and H. Xiong, "Multi-objective optimization based allocation of heterogeneous spatial crowdsourcing tasks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 7, pp. 1637–1650, Jul. 2018.
[27] M. H. Cheung, R. Southwell, F. Hou, and J. Huang, "Distributed time-sensitive task selection in mobile crowdsensing," in *Proc. ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2015, pp. 157–166.
[28] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: A load-balancing solution," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2092–2104, Feb. 2020.
[29] Z. Yu, Y. Gong, S. Gong, and Y. Guo, "Joint task offloading and resource allocation in UAV-enabled mobile edge computing," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3147–3159, Apr. 2020.
[30] T. Saaty, *The Analytic Hierarchy Process*. New York, NY, USA: Mcgraw-Hill, 1980.
[31] Z. Wang, J. Hu, J. Zhao, D. Yang, H. Chen, and Q. Wang, "Pay on-demand: Dynamic incentive and task selection for location-dependent mobile crowdsensing systems," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst.*, 2018, pp. 611–621.
[32] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Berlin, Germany Springer, 2004.

Yang Huang received the B.E. degree in automation in 2019 from the China University of Petroleum, Beijing, China, where he is currently working toward the M.E degree in control science and engineering with the College of Control Science and Engineering.

His research interest includes mobile crowdsensing.

Honglong Chen (Senior Member, IEEE) received the Ph.D. degree in computer science from The Hong Kong Polytechnic University, Hong Kong, in 2012.

From 2015 to 2016, he was a Postdoctoral Researcher with the School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Phoenix, AZ, USA. He is currently an Associate Professor and a Ph.D. Supervisor with the College of Control Science and Engineering, China University of Petroleum, Beijing, China. His current research interests include the Internet of Things and cybersecurity.

Guoqi Ma received the B.E. degree in automation in 2019 from the China University of Petroleum, Beijing, China, where he is currently working toward the M.E. degree in control science and engineering with the College of Control Science and Engineering.

His research interest includes mobile crowdsensing.

Kai Lin received the B.E. degree in automation from Tiangong University, Tianjin, China, in 2015, and the M.E. degree in control engineering in 2019 from the China University of Petroleum, Beijing, China, where he is currently working toward the Ph.D. degree in control science and engineering with the College of Control Science and Engineering.

His current research interests include RFID technology and the Internet of things.

Zhichen Ni received the B.E. degree in measurement and control technology and instrumentation in 2019 from the China University of Petroleum, Beijing, China, where he is currently working toward the M.E. degree in control science and engineering with the College of Control Science and Engineering.

His current research interests include edge computing and edge intelligence.

Na Yan received the B.E. degree in automation in 2019 from the China University of Petroleum, Beijing, China, where she is currently working toward the M.E. degree in control science and engineering with the College of Control Science and Engineering.

Her research interest includes RFID.

Zhibo Wang (Senior Member, IEEE) received the B.E. degree in automation from Zhejiang University, Hangzhou, China, in 2007, and the Ph.D degree in electrical engineering and computer science from the University of Tennessee, Knoxville, TN, USA, in 2014.

He is currently a Professor with the Institute of Cyberspace Research, College of Computer Science and Technology, Zhejiang University. His research interests include AI security, Internet of Things, network security, and privacy protection.