Fair Energy-efficient Sensing Task Allocation in Participatory Sensing with Smartphones

Qingwen Zhao†, Yanmin Zhu†, Hongzi Zhu†, Jian Cao†, Guangtao Xue†, Bo Li‡
† Shanghai Jiao Tong University
‡ Hong Kong University of Science and Technology
† qwzhao@sjtu.edu.cn, {yzhu, hongzi, cao-jian, xue-gt}@cs.sjtu.edu.cn, ‡bli@cse.ust.hk

***Abstract*—With the proliferation of smartphones, participatory sensing using smartphones provides unprecedented opportunities for collecting enormous sensing data. There are two crucial requirements in participatory sensing, *fair task allocation* and *energy efficiency*, which are particularly challenging given high combinatorial complexity, tradeoff between energy efficiency and fairness, and dynamic and unpredictable task arrivals. In this paper, we present a novel fair energy-efficient allocation framework whose objective is characterized by *min-max aggregate sensing time*. We rigorously prove that optimizing the min-max aggregate sensing time is NP hard even when the tasks are assumed as a priori. We consider two allocation models: *offline allocation* and *online allocation*. For the offline allocation model, we design an efficient approximation algorithm with the approximation ratio of $2 - \frac{1}{m}$, where m is the number of member smartphones in the system. For the online allocation model, we propose a greedy online algorithm which achieves a competitive ratio of at most m. The results demonstrate that the approximation algorithm reduces over 81% total sensing time, the greedy online algorithm reduces more than 73% total sensing time, and both algorithms achieve over 3x better min-max fairness.**

I. Introduction

With the proliferation of mobile devices, participatory sensing with smartphones becomes a new and important paradigm for collecting and sharing data with the general public. A lot of collaborative, crowdsourcing based applications spring up. Example applications of participatory sensing include intelligent transportation [1] [2] [3], localization [4] [5] [6], environmental monitoring [7] [8], and crowding counting [9].

The typical architecture of a participatory sensing system is illustrated in Fig. 1. The system is comprised of a *central platform* and a *collection of smartphones*. The platform residing on the cloud accepts sensing service requests from system users, and allocates sensing tasks to the member smartphones. After being assigned a sensing task, a smartphone performs the required sensing service and returns the sensing data to the platform which forwards the data to the querying user. The smartphones in the participatory sensing system are assumed *cooperative (not strategic)*, which belong to or affiliated to the system, willing to take sensing tasks and provide sensing services to the system. We call such smartphones *member smartphones*. Such participatory sensing systems are practical and realistic in enterprise or agreement-based cooperation scenarios. The issue of participation incentive [10] [11] of rational or even strategic smartphone user is out of the scope of the paper.

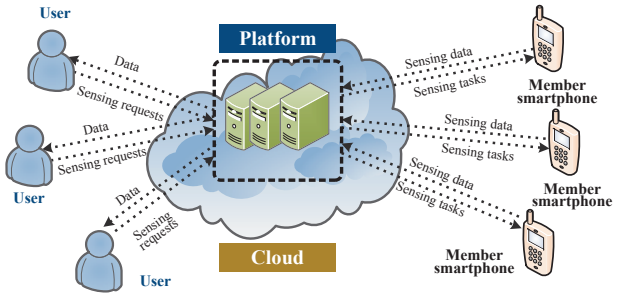

Fig. 1. The architecture of a participatory sensing system.

In this paper we focus on energy efficiency of member smartphones of a participatory sensing system. More specifically, we study the task allocation strategy of the platform for optimizing the energy efficiency of member smartphones. Processing a sensing task typically requires a smartphone to drive the processor for sampling and processing the data. It consumes considerable energy on the smartphone, which is dependent on the required sensing time length of a sensing task. Recent measurement study [12] has shown that the processor consumes up to 25% and an energy hungry sensor like GPS can consume up to 15% of the total energy. As a result, it causes large energy cost as a smartphone contributes to the participatory sensing system.

We make the important observation that it can greatly save energy consumption by reusing the sensing service on the

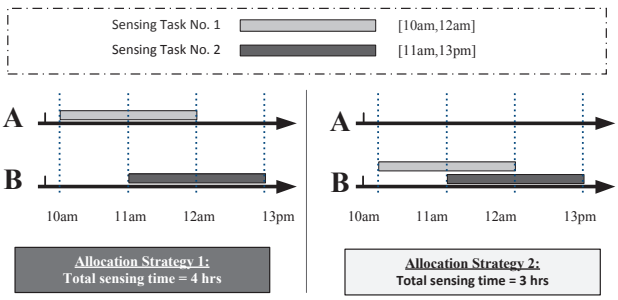

Fig. 2. Two different allocation strategies considered. The left figure considers the case two overlapped tasks are allocated to two different smartphones, and the total sensing time is 4 hours, while the right figure considers that two overlapped tasks are allocated to one smartphone, and thus the total sensing time 3 hours.

978-1-4799-3360-0/14/$31.00 ©2014 IEEE

smartphone, which is to allocate overlapping tasks requiring the same sensing service to the same smartphone. As illustrated by the example in Fig. 2, there are two sensing tasks: task 1 requests a sensing service from 10am to 12am, and task 2 requests the sensing service from 11am to 13pm. Suppose there are two smartphones A and B. We examine two strategies. In the first strategy, each smartphone is allocated one task. In this case, the total sensing time of both A and B is 4 hours. In the second strategy, both tasks are allocated to B. The resulting total sensing time is 3 hours. In conclusion, overlapping the sensing intervals of different tasks can reuse the sensing service on the smartphone and hence energy consumption can be reduced. Nevertheless, we also find that although the total energy is reduced, the issue of unfairness arises. It is clear that A spends no energy on sensing service while B spends 3 hours.

The previous observation motivates us to investigate the crucial problem of allocating sensing tasks for maximizing energy efficiency while maintaining good fairness among member smartphones. However, several great challenges remain to be solved. *First*, there is an intrinsic tradeoff between total energy efficiency and fairness among smartphones. It is highly desirable to strike a good balance between overall energy efficiency and fairness in terms of individual energy consumption. *Second*, both the number of sensing tasks and the number of smartphones can be large. The time complexity would be high if a straightforward exhaustive search is applied. We rigorously prove that the problem of task allocation for optimal min-max energy efficiency is *NP hard*. *Finally*, in the real world sensing tasks may arrive to the system at anytime and the arrival process of sensing tasks can be arbitrary and unknown beforehand.

Little existing work in recent years has studied the problem of task allocation in participatory sensing systems. In [13], the authors consider task assignment in a crowdsourcing market such as Amazon Mechanical Turk. The problem is to match heterogeneous tasks to workers with different, unknown skill sets. The objective is to maximize the total benefits of the requester who submitted tasks. In [14], the problem of selecting a service provider from a list of providers is considered, with the objective of maximizing the consumer's satisfaction. There is little existing work for task allocation, which is applicable to the problem of maximizing overall energy efficiency and fairness among smartphones. The unique characteristics of participatory sensing, such as tradeoff and utilization of overlapping intervals, have never been explored.

In response to the challenges mentioned above, we introduce a fair energy-efficient allocation framework whose objective is characterized by min-max aggregate sensing time. Based on the framework, we propose two sensing task allocation algorithms in participatory sensing systems for different allocation models: *offline allocation* and *online allocation*. For the offline allocation model, at the time of scheduling the platform has the complete knowledge of all sensing tasks, including the future tasks to be allocated. We design an efficient approximation algorithm with the approximation ratio of $2 - \frac{1}{m}$, where m is the number of member smartphones in the system. For the online allocation model, sensing tasks arrive to the system and the allocation decision is made on the fly. We propose a greedy algorithm with a polynomial time complexity, which achieves a competitive ratio of at most m.

The main technical contributions made in this paper are summarized as follows:

- It is *the first work*, to the best of our knowledge, that investigates the important problem of sensing task allocation in participatory sensing systems, with the objectives of achieving both energy efficiency and fairness.
- We introduce a novel fair energy-efficient allocation framework whose objective is characterized by *min-max aggregate sensing time*, and rigorously prove that the problem of optimizing min-max aggregate sensing time is NP hard even when tasks are known a *priori*.
- For the offline allocation model, we design an efficient approximation algorithm with the approximation ratio of $2-\frac{1}{m}$, where m is the number of member smartphones in the system. For the online allocation model, we propose a greedy algorithm with polynomial complexity of of $O(mn)$, which achieves a competitive ratio of at most m. n is the total number of tasks have allocated.
- We have performed both theoretical analysis and simulations, and the results demonstrate that, compared to the baseline algorithm, our algorithms achieve good energy efficiency and fairness. The approximation allocation algorithm reduces over 81% total sensing time, and the online allocation algorithm reduces more than 73% total sensing time. Both algorithms achieve over 3x better min-max fairness.

The rest of this paper is organized as follows. In Section II, we present the system model and define the problem. Section III describes the design of the approximation algorithm for the offline task allocation model and Section IV describes the design of the online algorithm for the online task allocation model. Section V presents and discusses evaluation results. Section VI reviews related work. In Section VII, we conclude the paper and present the future work directions.

II. System Model and Problem Formulation

In this section we first present the system model, and then formally formulate the problem considered in the paper. At last, we give the complexity analysis of the problem.

A. System Model

We consider sensing task allocation problem in a participatory sensing system with cooperative member smartphones. The platform is located on the cloud, accepting dynamically arriving tasks. The platform is responsible for allocating sensing tasks to the smartphones. There are m member smartphones in the system. A smartphone performs the task by sampling the required sensing data. On the completion of a sensing task, the smartphone returns the sensing data back to the platform which then forwards the data to the user who submitted the sensing task.

In this paper, we consider *homogeneous sensing tasks* which require the same sensing service from smartphones. Discussions on heterogenous tasks will be left for future studies. Each task r_i is associated with a sensing interval $I_i = [s_i, e_i]$, indicating that the sensing service starts from s_i and ends at e_i. The sensing time of task r_i is $e_i - s_i$. For ease of presentation, we consider the time as discrete slots of equal size (1 unit time). Note that a task can only be allocated to one smartphone and is indivisible.

Before formally defining the problem of the sensing task allocation, we first introduce some notations similar to [15].

Definition 1 (Cover of sensing tasks)**.** *For two tasks r_1, r_2 with sensing intervals $I_1 = [s_1, e_1]$ and $I_2 = [s_2, e_2]$ and $s_1 \leq s_2$, if $e_1 \geq e_2$, we call r_2 is covered by r_1.*

Definition 2 (Union of sensing intervals)**.** *Define $I_1 \uplus I_2$ as the union of two sensing intervals or interval sets. For example, $[2,5] \uplus [3,6] = [2,6]$, $\{[2,4],[5,7]\} \uplus [3,6] = \{[2,7]\}$.*

Definition 3 (Length of interval)**.** *Let $l(I)$ denote the length of interval I or the length of the union of intervals in set I. For example, $l([1,3] \uplus [2,5]) = l([1,5]) = 4$, $l([1,3] \uplus [4,5]) = 3$, and $l([1,4]) = 3$.*

Definition 4 (Aggregate sensing time)**.** *The* aggregate sensing time *ℓ_i of a smartphone i is the overall sensing time that i should spend on completing the allocated tasks. Given tasks allocated to i is $\{r_1, r_2, \cdots, r_n\}$ with intervals $\{I_1, I_2, \cdots, I_n\}$, then $\ell_i = l(\uplus_{i=1}^{n} I_i)$.*

B. Problem Formulation

We next formally define the task allocation problem whose objective is to optimize energy efficiency and maximize fairness. Since the two objectives are contradictory, we introduce a novel fair energy-efficient allocation framework whose objective is characterized by *min-max aggregate sensing time*. By achieving the min-max aggregate sensing time objective, we can jointly take fairness maximization and energy efficiency into consideration.

Definition 5 (Task allocation problem with the objective of min-max aggregate sensing time)**.** *Consider time period $[0, T]$ where T is a sufficiently large future time point of interest. During $[0, T]$, the set of sensing tasks that arrive to the system is denoted by $\mathcal{R} = \{r_1, r_2, \cdots, r_n\}$, with corresponding intervals $\mathcal{I} = \{I_1, I_2, \cdots, I_n\}$. The system has m member smartphones. The problem is to find a sensing task allocation, such that the maximum aggregate sensing time of smartphones is minimized, i.e.,*

$$\min \max_{1 \leq i \leq m} \ell_i, \tag{1}$$

We consider two task allocation models, i.e., offline allocation and online allocation.

Definition 6 (Offline task allocation model)**.** *In this model, at the time of scheduling the platform has the complete knowledge of all sensing tasks, i.e., both $\mathcal{R}$ and $\mathcal{I}$.*

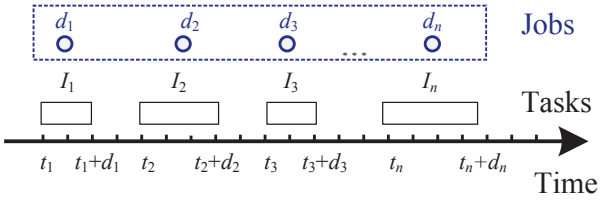

Fig. 3. Reduction from job scheduling to task allocation

Remarks: The offline task allocation model has limited applications in reality. We consider this model for studying the problem complexity and as a baseline for comparison with the online algorithm to be proposed.

Definition 7 (Online task allocation model)**.** *In this model, the platform makes the allocation decision once the task arrives to the system. The platform has no access to the knowledge of future tasks and their corresponding intervals.*

Remarks: This model is practical and applicable to real-world participatory sensing systems.

C. Analysis of NP Hardness

Optimizing the objective of min-max aggregate sensing time in a participatory sensing system is computationally difficult. In this subsection, we rigorously prove that the task allocation with the objective of optimizing the min-max aggregate sensing time is NP-hard even under the offline model.

Theorem 1. *The task allocation problem with the objective of minimizing the maximum aggregate sensing time of all smartphones under the offline allocation model is NP-hard.*

Proof: We prove the NP hardness by reducing from a classical NP problem of job scheduling [16] to our problem. The job scheduling problem can be described as follows: a sequence of jobs need to be scheduled on m identical parallel machines. Each job has a processing time. The goal is to find a optimal schedule which minimizes the makespan, which is the total processing time of all jobs scheduled on the most loaded machine.

This reduction takes an instance of the job scheduling problem as input. Given a set of jobs $\mathcal{J}$ and m identical parallel machines, each job j_i has a processing time d_i. Order these jobs arbitrarily, such as $\mathcal{J} = \{j_1, j_2, \cdots, j_n\}$, we construct an instance of the sensing task allocation problem as follows: for each job j_i, there is a sensing task with interval $I_i = [t_i, t_i + d_i], d_i > 0$. At the same time, $t_i + d_i < t_{i+1}, i \in \{1, 2, \cdots, n-1\}$. The reduction can be completed in polynomial time. An example is shown in Fig. 3

A job schedule with the makespan minimized can be translated into a task allocation with the maximum aggregate sensing time minimized. As there is no overlap between any two sensing intervals in this instance of task allocation problem, the sensing time of a task r_i is exactly equal to the processing time d_i of job j_i . Thus, the makespan of an optimal

scheduling is just the maximum aggregate sensing time of smartphones. ■

III. Approximate Offline Task Allocation

In this section we consider the offline allocation model. First, we present the overview of the offline task allocation algorithm, and next give the algorithm details and finally present some theoretical analysis.

A. Overview

As previously proved, the task allocation problem with the objective of min-max aggregate sensing time is NP hard and thus there is no computationally efficient algorithm for deriving the optimal solution. In this section, we design a polynomial-time approximation algorithm for the offline task allocation model under which the platform has the complete knowledge of tasks $\mathcal{R}$, including the sensing interval I_i of each task $r_i \in R$.

There are two steps in the design of the algorithm. In the *first* step, we construct a *task precedence graph* $G(V \cup \{v_0\}, E)$ with $|V| = n$. Being a directed graph, a task precedence graph is used to characterize the timing and inter-cover relations between sensing tasks. In the *second* step, we search for m paths which start from v_0. These m paths visit all the nodes in the graph collectively. Each node in V in the graph can only be included in a path exactly once. We call the set of such m paths *m-path*.

With this graph, we are able to convert the original problem of minimizing the maximum aggregate sensing time to a new problem of finding paths on a directed graph which is easier to handle.

B. Constructing Task Precedence Graph

We next explain the construction of the task precedence graph. Before introducing the construction of the graph, we introduce an important observation, which is given in the following claim.

Claim 1. *The optimal solution to the task allocation problem with the objective of min-max aggregate sensing time remains the same after any task which is covered by another task is removed from consideration.*

Remarks: Suppose that task r_i is covered by r_j and r_i is removed. The optimal solution with the new set of tasks without r_i remains optimal after r_i is added to the system because it can be allocated to the same smartphone which r_j is allocated to. It is clear that after r_i is added back, the optimal aggregate sensing time does not change. Therefore, we can remove those tasks which are covered by other tasks from consideration and the platform only allocate the new set of tasks. After the allocation is done, the removed tasks then are allocated into the corresponding smartphones.

Next we show how to convert a set of tasks in which no task is covered by another task, into a task precedence graph. Formally, we construct a directed graph $G(V \cup \{v_0\})$. Each node $v_i \in V$ represents a sensing task r_i and is attached with an interval $[s_i, e_i]$, where s_i is the start time and e_i is the end time. v_0 is an added virtual node with interval $[0, 0]$. There exists a directed edge $(v_i \to v_j) \in E$ if and only if $s_i < s_j$. The edge weight w_{ij} is the additional sensing time needed to complete task r_j after completing task r_i, which can be calculated as

$$w_{ij} = \begin{cases} l(I_i \uplus I_j) - l(I_i), & i \neq j \\ 0, & i = j \end{cases} \quad (2)$$

As task r_j ends later than task r_i, we have $w_{ij} > 0$.

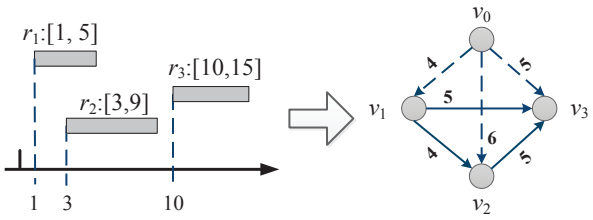

Fig. 4. The construction of the task precedence graph. Each node in the graph denotes a task with a start time and an end time. v_0 is the virtual node and denotes a task with the sensing interval [0,0]. The weight is calculated as definition 2

We give a simple example to show the constructing process of the task precedence graph (Fig. 4). Three tasks r_1, r_2, r_3 are converted to the nodes v_1, v_2, v_3. v_0 is a virtual node with the interval $[0, 0]$. A node with a small start time has an edge to a node with large start time. Thus there is an edge from v_0 to v_1, v_2, v_3, respectively. The weight of edge $(v_i \to v_j)$ is calculated as $l(I_i \uplus I_j) - l(I_i)$. For example, $w_{01} = 4 - 0 = 4$.

Claim 2. *An allocation with the maximum aggregate sensing time of smartphones minimized corresponds to a solution to finding m-path with the objective of minimizing the maximum length of those m paths.*

C. Searching for m-path

In this section we propose an approximation algorithm to search for m-path. The goal is to minimize the maximum length of the paths.

The algorithm for searching for paths proceeds in *two key steps*.

- In the *first* step, we search for a Hamilton path, denoted by P^*, from the constructed task precedence graph.
- In the *second* step, we split the obtained Hamilton path into m sections, each section corresponding to the set of tasks for a smartphone.

1) Searching for a Hamilton Path: We refer to a Hamilton path in a directed graph as a directed path that goes through each node exactly once, and a path is presented as a sequence of nodes in the remaining part of this paper.

A straightforward method to find the Hamilton path is to enumerate all paths from node v_0 to other nodes in the graph. Due to the characteristics of the task precedence graph, one can find the Hamilton path by the branch and bound algorithm which may have a fast convergent rate. A "branch" can be

Algorithm 1: Approximation algorithm

Input: The Hamiltonian path P^* and its length L_1^*; d_{max}; the number of smartphones, m
Output: a set of paths $\{P_1, P_2, \cdots, P_m\}$;
1: **for** each $j, 1 \leq j < m$ **do**
2: find the last node $v_{p(j)}$ such that the distance from v_0 to $v_{p(j)}$ along P^* is not greater than $\frac{j(L_1^*-d_{max})}{m} + d_{max}$.
3: Obtain the jth section denoted by a sequence P_j.

$$P_j = \begin{cases} < v_0, \ldots, v_{p(1)} >, & j = 1 \\ < v_0, v_{p(j-1)+1}, \ldots, v_{p(j)} >, & 1 < j < m \end{cases}$$

4: **end for**
5: $P_m = < v_0, v_{p(m-1)+1}, \ldots, v_n >$
6: return $P_1, P_2, \cdots, P_m$.

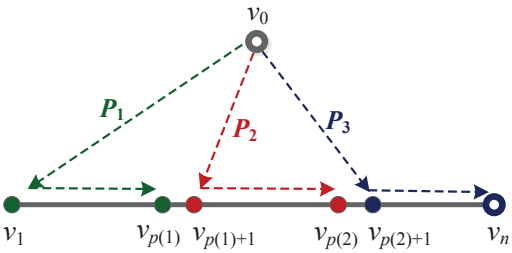

Fig. 5. An example of path splitting. A Hamiltonian path $v_0 \to v_1 \to \cdots \to v_n$ is split into 3 sections P_1, P_2, P_3. P_1 consists of $v_0, v_1, \cdots, v_{p(1)}$. P_2 consists of $v_0, v_{p(1)+1}, \cdots, v_{p(2)}$, and P_3 consists $v_0, v_{p(2)+1}, \cdots, v_n$.

pruned if there exists a node unvisited which has a smaller start time than those visited nodes on the branch. In this way, quite a lot of searching branches can be pruned. This is true because there is no possibility that a path revisits a node with a smaller start time.

The length of the Hamilton path, denoted by L_1^*, can be calculated by adding all the weights of the edges of the path.

2) Splitting the Hamilton Path: Next, we describe an approximation algorithm which employs a path splitting heuristic. Given the Hamilton path P^*, the algorithm splits it into m sections, $\mathcal{P} = \{P_1, P_2, \cdots, P_m\}$. Each section is built by a subcomponent of P^* with v_0 added as the source node. For ease of exposition, let d_{max} denote the maximum distance from v_0 to other nodes in V. Formally we have $d_{max} = \max_{1 \leq i \leq n} w_{0i}$. The details of the algorithm are given in Algorithm 1

For each section P_i, the distance from the first node v_0 to the last node $v_{p(i)}$ is actually the aggregate sensing time of corresponding tasks. The nodes in section $P_1, P_2, \cdots, P_m$ returned by the algorithm is the allocation scheme for the tasks. To ease the understanding, an example of splitting a path into 3 sections is shown in Fig. 5.

D. Optimization

Constructing the task precedence graph requires $O(n^2)$ and searching for a Hamilton path in a directed graph is not in polynomial time [17]. Fortunately, we find that a Hamilton path can be constructed simply by sorting all the tasks according to the start time in an increasing order. Inspired by this observation, we propose an optimization for finding the Hamilton path, which has a low complexity of $O(n \log n)$.

Claim 3. *There is only one Hamilton path in the task precedence graph, which starts from v_0 to the node with the largest end time.*

As the directed edge $(v_i \to v_j) \in E$ exists when the start time of v_i is smaller than that of v_j, the Hamilton path must go through all nodes in the order sorted by the start time, and v_0 is the first node of the Hamilton path. Thus one can sort all the tasks by the start time in an increasing order. Then, these tasks comprise the nodes of the Hamilton path. The weights of edges on the Hamilton path are calculated as Definition 2. The total time complexity is $O(n \log n)$, where n is the number of tasks.

E. Analysis

Theorem 2. *Suppose λ is the maximum aggregate sensing time of smartphones achieved by Algorithm 1, and λ^* is the maximum aggregate sensing time achieved by an optimal allocation. Then we have*

$$\frac{\lambda}{\lambda^*} \leq 2 - \frac{1}{m} \tag{3}$$

where m is the number of smartphones.

Proof: From the algorithm, we can see that the distance from v_0 to $v_{p(1)}$ along P^* is no greater than $(L_1^* - d_{max})/m + d_{max}$. For each section j, $1 < j \leq m-1$, the distance from $v_{p(j-1)+1}$ to $v_{p(j)}$ is no greater than $(L_1^* - d_{max})/m$. The distance from $v_{p(m-1)+1}$ to v_n is still $(L_1^* - d_{max})/m$. Thus for each section j, the maximum length is no greater than $(L_1^* - d_{max})/m + d_{max}$. Therefore,

$$\begin{aligned} \lambda &\leq (L_1^* - d_{\max})/m + d_{\max} \\ &\leq L_1^*/m + (1 - 1/m) d_{\max} \end{aligned} \tag{4}$$

Due to $d_{max} \leq \lambda^*$ and $\lambda^* \geq L_1^*/m$, then we have $\lambda \leq \lambda^* + (1 - 1/m)\lambda^* = (2 - 1/m)\lambda^*$ and $\frac{\lambda}{\lambda^*} \leq 2 - 1/m$. The proof is completed. ■

Theorem 3. *The time complexity of the approximation algorithm is $O(nlogn)$, where n is the total number of tasks.*

Proof: The time complexity of finding the Hamilton path and calculating the length of it is $O(n \log n)$, and the splitting process needs the time complexity of $O(n)$. Thus, the total time complexity of the algorithm is $O(n \log n)$. ■

IV. Online Task Allocation

In this section, we consider the online task allocation model. First, we present the overview of the online allocation algorithm. Next, the details of the algorithm design are presented. Finally, we provide theoretical analysis on the algorithm.

A. Overview

We propose a greedy algorithm, which allocates tasks to smartphones based on three basic rules.

- **Rule 1**: allocating a task to the smartphone if it can be covered by tasks which have already been allocated to that smartphone.
- **Rule 2**: allocating a task to the smartphone which has the smallest aggregate sensing time if the task were allocated to the smartphone.
- **Rule 3**: if multiple smartphones meet Rule 2, then allocate the task to the smartphone with the least increased sensing time.

The priority of these three rules decreases from the first one to the last one.

B. Algorithm Design

We present the detailed design of the greedy algorithm. Each smartphone maintains a list which stores the ordered tasks that have been allocated to it. A task with a smaller start time is ordered ahead of the one with a larger start time. The insertion of a new task can be completed in linear time. When there is an incoming task r_t, which denotes the t_{th} task that has arrived in the system, the algorithm first calculates the aggregate sensing time ℓ_i^t of each smartphone $i, \in \{0, 1, \cdots, m-1\}$, and obtains the increased sensing time Δ_i^t, and $\Delta_i^t = \ell_i^t - \ell_i^{t-1}$ of smartphone i if r_t is allocated to it. Then, the algorithm allocates the task to the right smartphone. The pseudocode is given in Algorithm 2.

The algorithm performs the task allocation following the three rules discussed above. If $\Delta_i^t = 0$, which means the incoming task can be covered by tasks allocated to smartphone i, then the algorithm allocates r_t to smartphone i. Otherwise, if smartphone i is the one with the smallest ℓ_i^t, the task r_t will be allocated to i, and then the algorithm returns. In the third case, there exists multiple smartphones have the smallest aggregate sensing time. For example, $\ell_i^t = \ell_j^t = \min \ell^t$, the algorithm further checks their increased sensing time Δ_i^t and Δ_j^t. If $\Delta_i^t < \Delta_j^t$, allocate r_t to smartphone i, otherwise to smartphone j.

C. Analysis

In this section, we first present the competitive analysis on the greedy algorithm. Next, we show its computation complexity.

An online algorithm A is called $\rho - competive$ if for a task sequence $\sigma =< r_1, r_2, \cdots, r_n >$,

$$A(\sigma) \leq \rho \cdot OPT(\sigma), \tag{5}$$

where $A(\sigma)$ is the maximum aggregate sensing time generated by A and $OPT(\sigma)$ is the maximum aggregate sensing time generated by an optimal allocation for σ.

Theorem 4. *The competitive ratio of the greedy algorithm is at most m for the online sensing task allocation problem, where m is the number of member smartphones.*

Proof: Let ℓ_i^* be the aggregate sensing time of smartphone i generated by the optimal offline algorithm, and L_m^* presents the maximum aggregate sensing time of the online allocation.

Algorithm 2: Greedy online allocation algorithm

Input: The set $\mathcal{R}_i$ of allocated tasks on each smartphone i; the current aggregate sensing time ℓ_i^{t-1} of each smartphone i; the incoming task r_t;
Output: The smartphone which task r_t should be allocated.

```
minload = INF;
store the minimum sensing time
for each smartphone i, i ∈ {0, 1, ··· , m − 1} do
    store the current aggregate sensing time ℓ_i^{t−1};
    calculate the aggregate sensing time of smartphone i after
    allocating r_t to it, ℓ_i^t;
    Δ_i^t = ℓ_i^t − ℓ_i^{t−1}
    if Δ_i^t = 0 then
        choice = i;
        break;
    else
        if ℓ_i^t < minload or ℓ_i^t = minload and Δ_i^t is smaller
        then
            choice = i;
            minload = ℓ_i^t;
        end if
    end if
    resume the aggregation sensing time to ℓ_i^{t−1}.
end for
return choice.
```

We have,

$$mL_m^* \geq \sum_i^m \ell_i^* \geq L_1^*, \tag{6}$$

where L_1^* is the total minimum aggregate sensing time of all the tasks.

On the other hand, we can easily conclude that the maximum aggregate sensing time L_m resulting from the greedy algorithm will never exceed L_1^*, i.e., $L_m \leq L_1^*$. By induction, $L_m \leq L_1^* \leq mL_m^*$ and $\frac{L_m}{L_m^*} \leq m$. ■

Theorem 5. *The time complexity of the greedy algorithm for per task allocation is $O(nm)$, where m is the number of smartphones and n is the number of sensing tasks that have been allocated so far.*

Proof: The calculation of the aggregate sensing time ℓ_i^t on each smartphone i during the allocation of task r_t can be completed in $O(n)$, and the total time complexity for calculating the aggregate sensing time of tasks on smartphones is $O(m \times n)$. Thus, the total time complexity of the greedy algorithm for per task allocation is $O(nm)$. ■

V. Performance Evaluation

A. Methodology and Simulation Setup

We use a *random* allocation algorithm as the baseline for performance comparison with both the offline and the online allocation algorithms. With the random allocation algorithm, a task is randomly allocated to one of member smartphones in the system.

Two metrics are used for performance evaluation, i.e., *min-max fairness* and *total sensing time*. Each smartphone has its aggregate sensing time, and the min-max fairness is the maximum aggregate sensing time. A smaller min-max fairness is desirable. The total sensing time is the sum of the aggregate sensing time of all smartphones in the system, which indicates

TABLE I
DEFAULT SETTINGS

Parameter	value
Number of tasks	400
Number of smartphones	30
Maximum length of intervals (min)	60

TABLE II
CONFIGURATION OF PARAMETERS

Cases	Number of tasks	Number of smartphones
Case 1	10	2
Case 2	15	2
Case 3	10	3
Case 4	12	3

the energy efficiency of the participatory system as a whole. A smaller total sensing time indicates better energy efficiency.

The default setting is as follows. We set the simulation time 24 hours for both the offline and the online algorithms. The time slot is one minute. Three impacting factors are investigated, i.e., number of tasks, number of smartphones and maximum length of intervals denoted by l_{max}. The length of the sensing interval of a task is chosen randomly in $[10, l_{max}]$ minutes. We assume the minimum length of intervals is 10 minutes, and the arrival of tasks obeys the Poisson process for both the offline and the online models. The default setting is summarized in Table I. Each data point is an average over 20 independent runs.

B. Impact of Number of Tasks

We first investigate the impact of the number of tasks on the performance. The number of tasks is changed from 200 to 700. The results are shown in Fig. 6 and Fig. 10.

In Fig. 6, one can find that as the number of tasks increases, the min-max fairness of smartphones becomes better. This is easy to understand because the total number of tasks to be executed becomes larger, the aggregate sensing time on each smartphone rises correspondingly. The approximation algorithm achieves the best min-max fairness, which is closely followed by the greedy algorithm. The random algorithm produces almost five times larger min-max fairness compared with the approximation algorithm when the number of tasks is 400. The increment of min-max fairness is not significant for both greedy algorithm and approximation algorithm as the number of tasks grows up. This shows our algorithms have good scalability. In Fig. 10, the total sensing time of all the smartphones increases with the number of tasks increasing. The total sensing time of the approximation algorithm is still the smallest. The greedy algorithm performs better than the random algorithm.

C. Impact of Number of Smartphones

To study the impact of the number of smartphones on the performance of algorithms, the second set of simulations varies the number of smartphones from 10 to 60. The results are shown in Fig. 7 and Fig. 11.

Fig. 7 shows that the min-max fairness decreases as the number of smartphones increases. The approximation algorithm achieves the best performance, and the greedy algorithm follows it. The random algorithm performs the worst. When the number of smartphones is 10, the min-max fairness of random algorithm is around five times as large as that of the approximation algorithm. The difference becomes smaller as the number of smartphones increases. In Fig. 11, with the increasing number of smartphones, the total sensing time increases. However, the approximation algorithm has the smaller increase rate of the total sensing time while the random algorithm has the largest increase rate. Because tasks are allocated to more smartphones, the overlap between intervals becomes smaller, and hence the total sensing time increases.

D. Impact of Maximum Length of Intervals

Finally, we study the impact of the length of intervals on the performance of algorithms. An intuition is that the more tasks with longer intervals, the larger overlap between two tasks. As a consequence, more sensing time is saved. In this set of simulations, the maximum length of sensing intervals is varied from 20 to 120 minutes with the increment of 20.

From Fig. 8 and Fig. 12, we can see that when the maximum length of intervals becomes larger, the min-max fairness of smartphones becomes larger. With the approximation algorithm, the min-max fairness increases by 80 when the maximum length of intervals changes from 20 to 120. With the random algorithm, the increase is as high as 700. The total sensing time, shown in Fig. 12, also increases with the increasing maximum length of intervals. Compared with the random algorithm, however, our algorithms have a much modest increase rate.

E. Comparison to the Optimal Solution

To show the efficiency of our algorithms, we compare the performance of our algorithms to the optimal solution derived from an exhaustive search algorithm. Four small-scale cases are designed, with the maximum length of intervals 40 minutes and the minimum length of sensing intervals 5 minutes. The total simulation time is 100 minutes. We change the number of tasks and the number of smartphones. The configuration of the parameters in the four cases is given in Table II.

The results are reported in Fig. 9 and Fig. 13. In Fig. 9, we can see that the min-max fairness is generally smaller when the number of smartphones becomes larger by comparing case 3, 4 with case 1, 2. As the number of tasks increases, the min-max fairness becomes worse. This also confirms the results we have observed in the previous simulations. Besides, the min-max fairness both for the greedy allocation algorithm and the approximation allocation algorithm is close to the optimum min-max fairness. The greedy allocation algorithm produces a similar min-max fairness as the approximation algorithm. This result is much better than what we expected, as the competitive ratio analyzed in Theorem 4 is at most m, where m is the number of member smartphones. In Fig. 13, the total sensing time achieved by the approximation algorithm closely follows the optimal total sensing time in all four cases. Although the

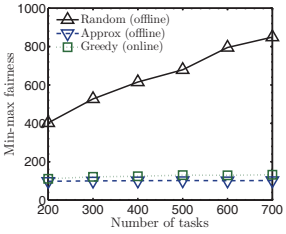

Fig. 6. Min-max fairness vs. number of tasks.

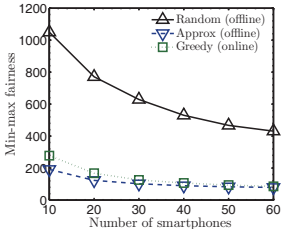

Fig. 7. Min-max fairness vs. number of smartphones.

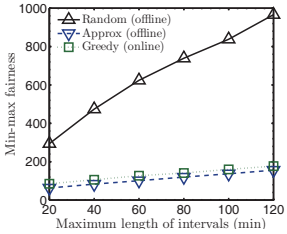

Fig. 8. Min-max fairness vs. maximum length of intervals.

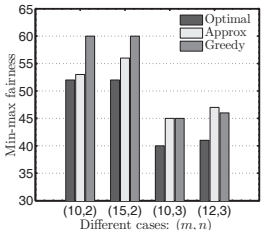

Fig. 9. Min-max fairness in different cases.

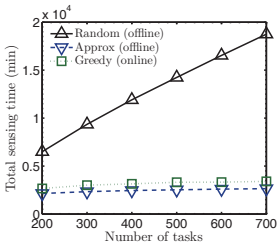

Fig. 10. Total sensing time vs. number of tasks.

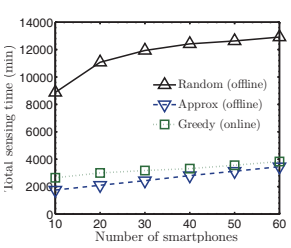

Fig. 11. Total sensing time vs. number of smartphones.

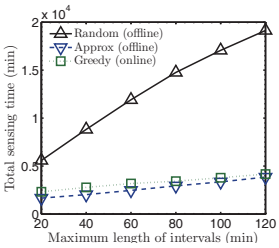

Fig. 12. total sensing time vs. maximum length of intervals.

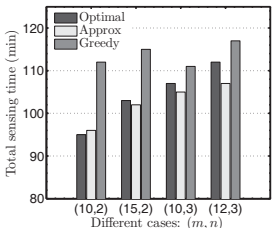

Fig. 13. Total sensing time in different cases.

total sensing time produced by the greedy allocation is slightly larger, it is at most 120% of the optimal value.

VI. Related Work

Participatory sensing has recently attracted extensive research attention from both industry and academic due to its attractive applications [3] [7] [8] [18]. Much existing work has been done to address various kinds of participatory sensing issues, such as the privacy problem [19]–[22] and the incentive mechanism design [10] [11] [23]. This paper focuses on the sensing task allocation in participatory sensing systems.

Recently, there are a few work on data sharing have been proposed in wireless sensor networks (WSNs) [15] [24]. In [24], the problem of data sharing among multiple applications is discussed. This work assumes each application needs to sample discrete data at some time points, and these data can be shared by multiple applications. The work proposed in [15] considers a continuous interval of sampling data. The overlapped interval of data can also be shared by applications. The goal is to minimize the total sampling time for completing all the sensing tasks. However, they only consider the sampling optimization in the view of a node, instead of the whole the wireless sensor networks, such as the load balancing problem of energy consumption.

Quite a lot of works on job/task assignment which aim to achieve load balancing have been proposed [25]–[28], both in offline and online cases. Most of them assume the multiple jobs cannot be performed on a machine concurrently [27] [28], i.e., there is only a job run at a time. In addition, theoretically, a job can be performed at any time, which is different from the task model defined in this paper. Although there also exist works which discuss the assignment of temporary tasks which have limited duration in time [26] [29], they also assume no more than one job/task can be executing at any time.

Some existing work has studied the problem of task allocation in crowdsourcing markets [13] [14] [30], and much of them consider how to maximize the benefits obtained by service requesters. In [13], the authors consider how to assign heterogeneous tasks to workers with different, unknown skill sets in the crowdsourcing markets such as Amazon Mechanical Turk. Given a fixed set of tasks and the times of each task need to be completed, and workers arrive online and one at a time, the goal is to allocate the workers to tasks such that the total benefit that the requester obtained maximized. A two-phase exploration-exploitation assignment algorithm is presented, which is proved to be competitive with respect to the optimal offline algorithm which knows the skill levels of each worker. The problem considered in [14] is to select a service provider from a list of providers which can provide maximum satisfaction to the service requester. An adaptive task scheduling which based on the customer satisfaction feedbacks is proposed. In [30], Ho et al. investigate the task assignment and label inference for heterogenous classification tasks. Labels are provided for instances (such as “websites”) by workers. By applying online primal-dual techniques, a near-optimal adaptive assignment algorithm is derived. In [10], T. Luo and C. Tham link incentive to users’ demand for consuming services. The problem is to assign an amount of service quota to users with the objective of maximizing fairness or social welfare.

In summary, little existing work has studied the problem of sensing task allocation in participatory sensing systems with the objective of maximizing energy efficiency of smartphones and fairness among smartphones.

VII. Conclusion and Future Work

In this paper we have studied the task allocation which is of paramount importance to both energy efficiency and fairness among smartphones. We have rigorously proven that the task allocation problem of minimizing the maximum aggregate sensing time is NP hard even under the offline model. We consider two task allocation models including offline model and online model. Under the offline allocation model, we have designed a polynomial-time approximation algorithm that approximates the offline optimum within a small factor of $2\text{-}\frac{1}{m}$, where m is the number of smartphones in the system. Under the online allocation model, we have designed a polynomial-time greedy algorithm that achieves a competitive ratio of at most m. We have presented theoretical analysis for both the offline algorithm and the online algorithm. We have also conducted extensive simulations and comparison of different allocation algorithms. The results demonstrate our algorithms can achieve high energy efficiency while keeping good fairness among smartphones.

In our future work, we will explore the following directions. *First*, sensing task can be heterogeneous, differing in required sensors, host operating systems and etc. This suggests that a task can only be allocated to a subset of the smartphones. In addition, two different events may require different hardware or components on the smartphone to process them. Although two different events have overlapping intervals, the overlapping porion may consume additional power. We shall study the impact of heterogeneous events. *Second*, currently we have focused on the reduction of sensing time on smartphones and no energy saving has been measured. We shall include prototype based measurements for energy consumption savings. *Finally*, events with flexible starting and ending times will be investigated, and our work will be extended to support such events.

References

[1] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, "Cooperative transit tracking using smart-phones," in *Proc. ACM SenSys*, 2010.

[2] N. Pham, R. K. Ganti, Y. S. Uddin, S. Nath, and T. Abdelzaher, "Privacy-preserving reconstruction of multidimensional data maps in vehicular participatory sensing," in *Wireless Sensor Networks*. Springer, 2010, pp. 114–130.

[3] K. Ali, D. Al-Yaseen, A. Ejaz, T. Javed, and H. S.Hassanein, "Crowdits:crowdsourcing in intelligent transportation systems," in *IEEE WCNC*, 2012.

[4] A. Rai, K. K. Chintalapudi, V. N.Padmanabhan, and R. Sen, "Zee: Zero-effort crowdingsourcing for indoor localization," in *Proc. ACM SIGOMM*, 2012.

[5] J. Zhu, K. Zeng, K.-H. Kim, and P. Mohapatra, "Improving crowdsourced wi-fi localization systems using bluetooth beacons," in *IEEE SECON*, 2012.

[6] F. Zaid, D. Costantini, P. S.Mogre, A. Reinhardt, J. Schmitt, and R. Steinmetz, "Wbroximity:mobile participatory sensing for wlan-and bluetooth-based positiong," in *IEEE LCN*, 2010.

[7] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: an end-to-end participatory urban noise mapping system," in *Proc. ACM IPSN*, 2010.

[8] J. Ballesteros, M. Rahman, B. Carbunar, and N. Rishe, "Safe cities.a participatory sensing approach," in *IEEE LCN*, 2012.

[9] P. G. Kannan, S. P. Venkatagiri, M. C. Chan, A. L. Ananda, and L.-S. Peh, "Low cost crowd counting using audio tones," in *Proc. ACM SenSys*, 2012.

[10] T. Luo and C.-K. Tham, "Fairness and social welfare in incentivizing participatory sensing," in *IEEE SECON*, 2012.

[11] Y. Zhang and M. v. d. Schaar, "Reputation-based incentive protocols in crowdsourcing applications," in *Proc. IEEE INFOCOM*, 2012.

[12] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely, "Energy-delay tradeoffs in smartphone applications," in *Proc. ACM MobiSys*, 2010.

[13] C.-J. Ho and J. W. Vaughan, "Online task assignment in crowdsourcing markets," in *AAAI*, 2012.

[14] V. Nunia, B. Kakadiya, C. Hota, and M. Rajarajan, "Adaptive task scheduling in service oriented crowd using slurm," in *Distributed Computing and Internet Technology*. Springer Berlin Heidelberg, 2013, vol. 7753, pp. 373–385.

[15] X. Fang, H. Gao, J. Li, and Y. Li, "Application-aware data collection in wireless sensor networks," in *Proc. IEEE INFOCOM*, 2013.

[16] M. R. Garey and D. S. Johnson, "Strong NP-Completeness Results: Motivation, Examples, and Implications," *Journal of the ACM*, vol. 25, no. 3, pp. 499–508, 1978.

[17] G. Narasimhan, "A note on the hamiltonian circuit problem on directed path graphs," *Information Processing Letters*, vol. 32, no. 4, pp. 167–170, 1989.

[18] H. Weinschrott, J. Weisser, F. Durr, and K. Rothermel, "Participatory sensing algorithms for mobile object discovery in urban areas," in *IEEE percom*, 2011.

[19] K. Xing, Z. Wan, P. Hu, H. Zhu, Y. Wang, X. Chen, Y. Wang, and L. Huang, "Mutual privacy-preserving regression modeling in participatory sensing," in *Proc. IEEE INFOCOM*, 2013.

[20] H. Ahmadi, N. Pham, R. Ganti, T. Abdelzaher, S. Nath, and J. Han, "Privacy-aware regression modeling of participatory sensing data," in *Proc. ACM SenSys*, 2010.

[21] K. Vu, R. Zheng, and J. Gao, "Efficient algorithms for k-anonymous location privacy in participatory sensing," in *Proc. IEEE INFOCOM*, 2012.

[22] F. Zhang, L. He, W. He, and X. Liu, "Data perturbation with state-dependent noise for participatory sensing," in *Proc. IEEE INFOCOM*, 2012.

[23] J.-S. Lee and B. Hoh, "Sell your experiences: A market mechanism based incentive for participatory sensing," in *IEEE percom*, 2009.

[24] A. Tavakoli, A. Kansal, and S. Nath, "On-line sensing task optimization for shared sensors," in *Proc. ACM IPSN*, 2010.

[25] N. Buchbinder and J. S. Naor, "Fair online load balancing," in *Proc. ACM SPAA*, 2006.

[26] Y. Azar, B. Kalyanasundaram, S. Plotkin, K. R. Pruhs, and O. Waarts, "Online load balancing of temporary tasks," in *Algorithms and Data Structures*. Springer, 1993.

[27] J. Fakcharoenphol, B. Laekhanukit, and D. Nanongkai, "Faster algorithms for semi-matching problems," in *Automata, Languages and Programming*. Springer, 2010, pp. 176–187.

[28] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts, "On-line load balancing with applications to machine scheduling and virtual circuit routing," in *Proc. ACM STOC*, 1993.

[29] J. Xu and D. L. Parnas, "Scheduling processes with release times, deadlines, precedence and exclusion relations," *Software Engineering, IEEE Transactions on*, vol. 16, no. 3, pp. 360–369, 1990.

[30] C.-J. Ho, S. Jabbari, and J. W. Vaughan, "Adaptive task assignment for crowdsourced classification," in *ICML*, 2013.