Heterogeneous Task Allocation in Participatory Sensing

Fan Yang*, Jia-Liang Lu*, Yanmin Zhu*, Jia Peng*, Wei Shu*† and Min-You Wu*
*Dept. of Computer Science & Engineering, Shanghai Jiao Tong University, Shanghai, China
†Dept. of Electrical & Computer Engineering, University of New Mexico, USA
Email: {xcyangfan, jialiang.lu, pengjia2049, mwu}@sjtu.edu.cn, yzhu@cs.sjtu.edu.cn, shu@ece.unm.edu

***Abstract*—The proliferation of smartphones has enabled a novel paradigm, participatory sensing, which leverages the smartphones to collect and share data about their surrounding environment. Since the sensing tasks are location-dependent and have time features, it is crucial and challenging to find a proper allocation of sensing tasks to ensure the timeliness of tasks and the quality of sensing data. In this paper, we investigate the heterogeneous sensing task allocation problem aiming at minimizing the total penalty caused by the tardiness of tasks. We prove this problem is NP-hard and propose two hybrid algorithms which combine a heuristic algorithm and two metaheuristic algorithms respectively. The extensive simulation results show that the proposed hybrid algorithms outperform the metaheuristic algorithms.**

I. Introduction

With recent advances in wireless communication and Micro Electro Mechanical Systems (MEMS), smartphones have become more and more pervasive. Integrated with a variety of sensors (*e.g.* accelerometer, gyroscope, microphone, camera), smartphones are able to sense the surrounding environment. Being with a mobile user almost round-the-clock, a smartphone becomes an important bridge connecting the physical world to the cyber world. These advances have enabled a new promising paradigm, participatory sensing or crowdsensing, which harnesses the smartphones to collect and share data [1][2].

Compared with traditional sensing systems such as wireless sensor networks (WSNs), a participatory sensing system has many advantages. It is easier for participatory sensing to enable various applications (social networks [3][4], indoor localization [5][6], environmental monitoring [7][8] and traffic monitoring [9][10]) at a community scale with higher coverage/scalability and lower installation/maintenance cost [1][11]. Hence the research of this area is of importance.

Currently, various participatory sensing systems have been proposed for specific sensing applications. It is very interesting and helpful to build a unified architecture to support multiple applications, since most of them share common data requirements and similar functionalities [2]. One key challenge is the allocation of sensing tasks among smartphones. As illustrated in Fig.1, a typical participatory sensing system supporting various applications is composed of a central cloud platform, a collection of service requesters and a pool of smartphones. The cloud platform accepts sensing tasks from service requesters, then allocates those tasks to the smartphones. After performing the assigned tasks, the smartphone submits the sensing data to the cloud platform which runs aggregate analysis and forwards the result to the corresponding service requester.

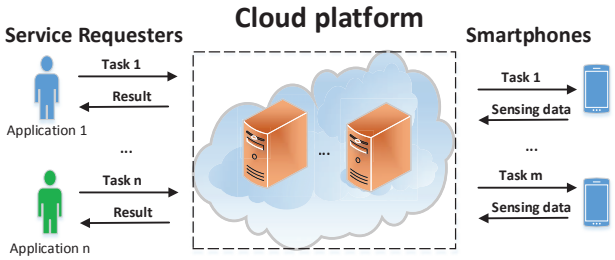

Fig. 1. The architecture of a unified participatory sensing system

Due to the unique characteristics of tasks and mobile users, it is crucial and challenging to assign tasks among mobile users to ensure the timeliness of tasks and the quality of sensing data. *First*, each task is location-dependent, which means it needs to be performed at a certain location, *e.g.* monitoring noise level at a specific location. *Second*, sensing tasks can have different task release times, deadlines and processing times. For example, in a traffic monitoring application, a task must be performed between 8 a.m and 9 a.m, and kept monitoring for 30 minutes to evaluate traffic congestion in a morning rush hour. *Third*, the mobile users also have different initial locations. Therefore, it will take some time for the mobile user to travel different places to finish tasks. *Fourth*, tasks should be processed within the given release times and deadlines. The deadlines are soft deadlines. If a task is finished later than its deadline, the quality of sensing data degrades accordingly.

Some researchers have studied the task allocation problem in participatory sensing. However, they both have some constraints. Zhao *et al.* [12] investigate the homogeneous sensing task allocation problem which neglects that tasks are location-dependent. He *et al.* [13] study the location-dependent sensing task allocation problem. However, they do not consider the time features of tasks.

In this paper, we study the heterogeneous sensing task allocation problem. Some challenges need to be solved. *First*, the completion time of each task depends on the task release time, processing time and the travelling time needed by a

978-1-4799-5952-5/15/$31.00 ©2015 IEEE

mobile user. A small change in the task allocation can result in finishing tasks with long tardiness. Thus, it is difficult to assign tasks to smartphones to complete them as early as possible, while ensuring the quality of contributed data. *Second*, there can be a large number of tasks and users. The complexity of an exhaustive search is high, and we prove this problem is NP-hard. Our main contributions are summarized as follows:

- It is the first work that investigates the heterogeneous sensing task allocation problem in participatory sensing systems, considering the spatial and temporal constraints in both tasks and mobile users. The objective is *minimizing the total penalty caused by tardiness of sensing tasks* to ensure the quality of data;
- We formally formulate this problem and rigorously prove it is NP-hard. We propose two hybrid algorithms to solve this problem and compare their performance. The hybrid algorithms combine a heuristic algorithm with two meta-heuristics respectively. We have conducted extensive simulations to evaluate their performance. The result shows that the proposed hybrid algorithms outperform the meta-heuristics.

The rest of this paper is organized as follows. Section II reviews some related work. In Section III, we introduce the system model and formulate the problem. Section IV describes the design of the proposed hybrid algorithms. Section V presents and discusses evaluation results. In Section VI, we conclude the paper and present the future work directions.

II. Related work

Recently, participatory sensing has received great attention due to the proliferation of smartphones and its various applications [3–10]. Some existing work has studied issues such as participation incentive [14], privacy and security of users [15]. This work focus on the task allocation problem.

Some researchers have studied the task allocation problem in participatory sensing. In [12], Zhao *et al.* study the allocation of the homogeneous sensing tasks for optimizing the energy efficiency of member smartphones, where tasks are identical and can be assigned to arbitrary smartphones. They ignore the important features (*e.g.* location) of tasks and users. He *et al.* [13] take into account the location-dependent tasks and user's time budget. However, they neglect that tasks have time features, especially the task processing time. In [16], Wang *et al.* study the sensing task allocation problem from the perspective of spatiotemporal coverage ratio. They only consider regular participants (*e.g.* city buses, school buses) with a regular spatiotemporal moving pattern.

Some research work has studied the task allocation problem in crowdsourcing markets. In [17], Ho *et al.* propose a two-phase exploration-exploitation assignment algorithm to assign heterogeneous tasks to workers with unknown skill sets. The workers can arrival on-line. The goal is to maximize the profit of the service requesters. In [18], the authors investigate the problem of determining a group of workers for each incoming task. They assume the task is with a list of possible answers, which is different from the task model in participatory sensing.

Unlike previous work, we investigate the heterogeneous task allocation problem in participatory sensing with the goal of minimizing the total penalty caused by tardiness of tasks.

III. Problem formulation

In this section, we first present the system model, and then formally define the problem. At last, we prove the NP-hardness of this problem.

A. System Model

As shown in Fig.1, a participatory system consists of a cloud platform, a set of service requesters and a pool of mobile users (*i.e.* smartphone users). The service requesters can publish different sensing tasks in the cloud platform, which then allocates tasks to the mobile users. After finishing the assigned tasks, the mobile user uploads the sensing data to the platform which forwards the result to the corresponding service requester. We denote the set of registered mobile users by $\mathcal{U} = \{u_1, u_2, ..., u_m\}$, and the collection of sensing tasks by $\mathcal{T} = \{t_1, t_2, ..., t_n\}$. The mobile users are identical except for different initial locations. They have the same moving speed. The initial position of user u_l is defined as P_{u_l}.

The sensing tasks are location-dependent and have different time features. The task t_j is present as $t_j =< P_{t_j}, r_j, d_j, p_j >$, where P_{t_j} is the location of t_j, r_j is the release time of t_j , d_j denotes the deadline and p_j denotes the processing time needed to accomplish t_j. A mobile user has to travel to different locations associated with tasks. We define $Time_{lj}$ as the travelling time from P_{u_l} to P_{t_j}. Let s_{ij} denote the time needed by a mobile user to travel between task t_i and t_j and we assume $s_{ij} = s_{ji}$. We define C_j as the completion time of t_j and $T_j = \max\{0, C_j - d_j\}$ as the the tardiness of t_j. In this problem, we allow late tasks, *i.e.* the tardiness T_j can be larger than zero.

We assume that a task can only be assigned to one mobile user and is indivisible. This can be achieved in the cloud platform by dividing a sensing task into a large number of indivisible tasks which can be performed on a single smartphone. The mobile users are volunteers or member users of the system, and they are always available. Such assumptions are practical in enterprise or agreement-based cooperation scenarios [12]. The time is considered as discrete slots of equal size (1 unit time). Hence, the r_j, d_j, p_j, $Time_{lj}$ and s_{ij} are non-negative integers.

B. Problem formulation

Next, we formally define the heterogeneous sensing task allocation problem. In order to obtain better sensing data, the goal is to minimize the total penalty caused by the tardiness. We define the penalty $Cost_j$ of task t_j as follows:

$$Cost_j = \begin{cases} C_0 + \alpha \cdot (T_j)^{\beta}, & \text{if } T_j > 0 \\ 0, & \text{if } T_j = 0 \end{cases} \quad (1)$$

C_0, α and β are the penalty parameters: $C_0 \geq 0$, $\alpha \geq 1$ and $\beta \geq 1$.

For convenience, we define some decision variables x_{il}, $t_i \in \mathcal{T}, u_l \in \mathcal{U}$: $x_{il} = 1$ if the t_i is assigned to u_l; $x_{il} = 0$ otherwise. y_{ij}, $t_i, t_j \in \mathcal{T}$: $y_{ij} = 1$ if the t_i and t_j are assigned to the same mobile user and t_i is scheduled immediately before t_j; $y_{ij} = 0$ otherwise.

The heterogeneous sensing task allocation problem with the objective of minimizing the total penalty can be formulated as follows:

$$\min \sum_{j=1}^{n} Cost_j \tag{2}$$

s.t.

$$\sum_{i=1}^{n}\sum_{l=1}^{m} x_{il} = n \tag{3}$$

$$\sum_{l=1}^{m} x_{il} = 1, i = 1...n \tag{4}$$

$$r_i + p_i + \sum_{i=1}^{n} y_{ij} \cdot s_{ij} \leq C_j, j = 1...n \tag{5}$$

$$p_i + \sum_{i=1}^{n} y_{ij} \cdot (s_{ij} + C_i) \leq C_j, j = 1...n \tag{6}$$

$$x_{il} = \{0, 1\} \tag{7}$$

$$y_{ij} = \{0, 1\} \tag{8}$$

$$T_i = \max\{0, C_i - d_i\} \tag{9}$$

Equation (3) ensures that all the tasks are scheduled. Constraint (4) ensures that each task is scheduled only on one mobile user. Constraint (5) and (6) ensure that a mobile user can only start tasks after finishing the previous task and travelling the required distance between different tasks. Note that a mobile user cannot start a new task until the new task is released and he has completed the preceding task.

C. Analysis of the NP-hardness

Theorem 1. *The heterogeneous sensing task allocation problem with the objective of minimizing the total penalty of all tasks is NP-hard.*

Proof: We prove the task allocation problem is NP-hard by reducing it to one kind of Parallel Machine Scheduling (PMS) problems [19], *i.e.* $Pm|r_j, s_{ij}|\sum T$ [20], which is known to be NP-hard. In the $Pm|r_j, s_{ij}|\sum T$ problem, Pm indicates that there are m identical parallel machines. The jobs have different arrival time r_i, processing time, deadline and a sequence dependent setup time s_{ij} of processing job i immediately after job j. Each job can only be assigned once to one machine. The optimizing goal is to find the optimal schedule of the jobs that minimizes $\sum T$, the sum of the tardiness of all the jobs.

Given an instance of $Pm|r_j, s_{ij}|\sum T$, we can construct the instance of the sensing task allocation problem as follows: each job with time features is mapped to a task with corresponding time features, and each machine is mapped to a mobile user. A sequence dependent setup time s_{ij} is mapped to the travelling time between task t_i and t_j. We set the penalty parameters as $C_0 = 0, \alpha = 1, \beta = 1$. Therefore, the sum of the tardiness of all the jobs in an optimal scheduling is just the total penalty of all tasks. For any instance of $Pm|r_j, s_{ij}|\sum T$, a corresponding instance of the heterogeneous sensing task allocation problem is reduced to it in polynomial time, which ends the proof. ■

IV. Task Allocation Algorithms

In this section, we present two hybrid heuristic algorithms for the heterogeneous sensing task allocation problem. The hybrid heuristic algorithms combine the Earliest-Completion-Time (ECT) heuristic with other meta-heuristic algorithms, *i.e.* the genetic algorithm and the iterated greedy algorithm respectively.

A. Earliest-Completion-Time Heuristic Algorithm

In order to minimize the total penalty of tasks, the ECT heuristic detailed in Algorithm 1 is proposed. It is meant to assign tasks to mobile users with earliest completion times. This heuristic can be integrated with other meta-heuristics for determining initial solutions.

At the beginning of this algorithm, all the tasks are sorted by the non-decreasing order of their deadline. So the tasks with earlier deadline have higher priority to be assigned according to the EDD (earliest due date first) rule. Initially, none of tasks is assigned to user u_l, so the $\mathcal{T}_l$ is empty.

From line 3 to line 16 in Algorithm 1, we assign each task to a mobile user. We try to find a mobile user u_{l^*} who can finish the task t_i with the earliest completion time from line 6 to line 14. For an arbitrary mobile user u_l, a task t_i cannot start earlier than its release time r_i or until u_l finishes his latest task t_k. So the expected completion time of task t_i is $C_i = \max\{r_i, C_k\} + s_{ki} + p_i$. We calculate the C_i for all $l = 1...m$, and finally we assign the task t_i to u_{l^*} with the earliest completion time.

The time complexity of ECT is O($n \log n + mn$), where n is the number of tasks and m is the number of mobile users. Since the time complexity of sorting all tasks is O($n \log n$) and the main loop needs the time complexity of O(mn).

B. Hybrid Genetic Algorithm

Genetic algorithm (GA) is one of the well-known stochastic search algorithms [21], which is based on the principles of the evolution theory. We design a hybrid genetic algorithm (HGA) which combines the ECT with the genetic algorithm. The ECT provides good initial solutions for the genetic algorithm to reduce the blind search space. Three genetic operators, *i.e* crossover, mutation, and selection, are performed to explore the solution space. The detail of HGA is introduced as follows:

1) Solution coding: The first step in the genetic algorithm is to determine the representation of solution or chromosome. As shown in Fig.2, each chromosome consists of $(n+m-1)$ genes, *i.e* n task genes and $m-1$ separation genes marked with '*', to divide n genes into m subchromosomes [22][23]. Each subchromosome represents a task allocation of a mobile user $\mathcal{T}_l$. If there are no task genes between two consecutive separation genes, the corresponding mobile user of the second

Algorithm 1 The ECT heuristic algorithm

Input:
$\mathcal{T} = \{t_1, t_2, ..., t_n\}$: the sensing task set;
$\mathcal{U} = \{u_1, u_2, ..., u_m\}$: the mobile user set;
Output:
$\mathcal{T}_l$: tasks assigned to user u_l, $l = 1...m$.

Sort all the tasks by non-decreasing order of their deadline d_i. We assume that $d_1 \leq d_2 \leq ... \leq d_n$.
Set $\mathcal{T}_l \leftarrow \emptyset$ for $l = 1...m$
for $i = 1 \rightarrow n$ **do** ▷ For each task t_i in $\mathcal{T}$
  $l^* = 0$ ▷ The most suitable mobile user
  $C_{min} = INF$
  for $l = 1 \rightarrow m$ **do**
    if $\mathcal{T}_l \neq \emptyset$ **then**
      $k = Latest(\mathcal{T}_l)$ ▷ k is the latest task assigned to u_l
      $C_i = \max\{r_i, C_k\} + s_{ki} + p_i$
    else
      $C_i = r_i + Time_{li} + p_i$
    if $C_{min} \geq C_i$ **then**
      $C_{min} = C_i$, $l^* = l$
  $C_i = C_{min}$, $T_i = \max\{0, C_i - d_i\}$,
  Calculate $Cost_i$ according to Eq.(1)
  $\mathcal{T}_{l^*} = \mathcal{T}_{l^*} \cup \{i\}$ ▷ Assign task t_i to the user u_{l^*}

separation gene has no tasks. When there are no tasks genes after the last separation gene, the rest mobile user is not assigned tasks.

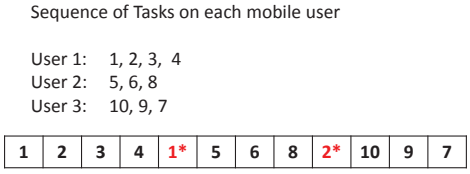

Fig. 2. The representation of chromosome

2) Initial population generation: The initial population is generated from two subpopulation with the same number of chromosomes [23]: one comes from the ECT, and the other is generated by randomly assigning all tasks to the mobile users. The size of the population is $PopSize$.

3) Crossover: According to a crossover probability P_c, all chromosomes in the parent population are mutually crossed over to generate the offsprings [21][24]. Fig.3 illustrates an example of the crossover operation.

- Choose two chromosomes, Parent1 and Parent2, from the parent population;
- Randomly generate a binary string of $n+m-1$ elements, filled with "1" or "0";
- Match the string with Parent1 such that genes corresponding to "1" are selected to the Part1 with the same positions in Parent1;
- Cross out the same genes in the Part1 from Parent2, and copy the left genes to the Part2;
- The offspring is generated by filling out the empty locations of Part1 with the genes of Part2 preserving the

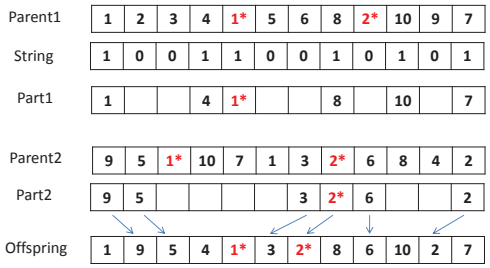

Fig. 3. An example of the crossover operation

4) Mutation: The mutation operator can generate new combinations of genes, which helps the search to jump out of a local optimum. We use the swap mutation according to a given mutation probability P_m as follows [21]: randomly select two genes from different subchromosomes and swap their positions. All the offsprings from the mutation are kept for the new generation.

5) Selection: Selection is an important operator to choose the proper chromosomes for the next generation. Since each chromosome represents a solution of task allocation, its fitness value is defined as objective value, *i.e.* total penalty, which can be calculated according to the Algorithm 2.

Algorithm 2 Calculate the total penalty of a solution of task allocation

Input:
$\mathcal{T}_l$: tasks assigned to user u_l, $l = 1...m$.
Output:
The total cost $\sum_{i=1}^{n} Cost_i$

for $l = 1 \rightarrow m$ **do**
  for take t_i from $\mathcal{T}_l$ according to its order **do**
    if t_i does not have a prior task **then**
      $C_i = r_i + Time_{li} + p_i$,
    else
      t_k is the direct prior task of t_i
      $C_i = \max\{r_i, C_k\} + s_{ki} + p_i$
    $T_i = \max\{0, C_i - d_i\}$, Calculate $Cost_i$ according to Eq.(1)
return The sum of $Cost_i$, $i = 1...n$

Let γ_i^g denote the selection value of chromosome i in the generation g. It can be calculated as

$$\gamma_i^g = (F_{max}^g - F_i^g)^2 \tag{10}$$

where F_{max}^g is the maximal fitness value of generation g. It is obvious that the lower the fitness value, the higher its selection value. Since our optimization goal is to minimize

the total penalty, the chromosomes with lower fitness values should have more chance to be selected. The most used selection mechanism is the "roulette wheel" sampling. The chance of chromosome being selected for the next generation is proportional to its selection value. The wheel is spun $PopSize$ times.

6) Stopping criteria: We use two rules to stop the HGA [24]: (1) the current generation number g is greater than the maximum number of generation G_{max}, and (2) the standard deviation of the fitness values of chromosomes in generation g is less than an arbitrary constant ε , which can be calculated as:

$$\sigma_g = \left[(\frac{1}{PopSize}) \sum_{i=1}^{PopSize} (F_i^g - \overline{F^g}) \right]^{1/2} \quad (11)$$

where $\overline{F^g}$ is the average fitness value of chromosomes in generation g.

C. Hybrid Iterated Greedy Algorithm

The iterated greedy (IG) heuristic is one of the meta-heuristics which has been applied successfully to a number of combinatorial optimization problems such as the large set-covering problem [25] and the parallel machine scheduling problem [26].

A general IG algorithm consists of 3 main steps: destruction, construction and acceptance. The hybrid iterated greedy algorithm (HIG) combines the ECT with the IG heuristic in order to avoid the blind search of IG at the beginning. The details of the hybrid algorithm are as follows:

1) Initial Solution Generation: The initial solution φ_0 is generated by the ECT algorithm. And we adopt the same solution representation as the HGA. The penalty of a solution can be calculated by the Algorithm 2. Set φ_0 as the current solution φ and the best solution φ_{best}.

2) Destruction: Randomly remove η tasks from the φ and save them in φ_p preserving the same order of selecting them. φ_d is the partial solution after the removal of η tasks.

3) Construction: Reinsert the tasks of φ_p into φ_d until a new full solution is constructed. The tasks of φ_p can be inserted into all possible positions in φ_d. The new best solution φ_{new} with minimal cost is recorded during the iterations.

4) Acceptance: We adopt a simulated annealing-like acceptance criterion with sinking temperature [26]. $\mathbf{T}_0$ is the initial temperature and T_{curr} is the current temperature. When the algorithm runs a fixed number of iterations I_{fixed}, T_{curr} will decrease as $T_{curr} = \lambda T_{curr}$, where $0 < \lambda < 1$.

If $\Delta Cost = Cost(\varphi_{new}) - Cost(\varphi_{best})$ is less than 0, we update the best solution $\varphi_{best} = \varphi_{new}$ and $\varphi = \varphi_{new}$. To jump out of a local optimum, we utilize the Boltzmann function which is used in the simulated annealing algorithms. This function starts with generating a random number $r \in [0, 1]$. If $r < e^{-\Delta Cost/T}$, we replace the φ with φ_{new}. Therefore, the penalty of φ can be improved iteratively through the destruction and construction phase.

5) Stopping criteria: The iterations do not stop until any of the following rules is met [26]: (1) the current temperature T_{curr} is less than a given temperature value T_{final}; (2) the reductions of temperature exceed a fixed number N_{max}.

V. Performance Evaluation

In this section, we implement both the meta-heuristic algorithms and hybrid algorithms, and evaluate their performance with the total penalty.

A. Simulation Setup

We assume a region of size 3000m×3000m. The sensing tasks and mobile users are randomly distributed in this region. The time slot is one minute. The release time r_j and the processing time p_j are randomly generated in $[0, 1000]$ and $[1, 30]$ respectively. And the deadline is set as $d_j = r_j + p_j + \Delta d$, where Δd is randomly chosen in $[0, 15]$. We set the moving speed of user is 4 km/h. Hence the s_{ij} and $Time_{lj}$ can be calculated through dividing the Euclidean distance between tasks and users by the moving speed. We set the penalty parameters as $C_0 = 10$, $\alpha = 1.25$ and $\beta = 2$. For each test case, we generate 20 instances and run 10 times for each instance. According to preliminary experiments, we determine the parameters for HGA are as follows: $PopSize = 20$, $P_c = 0.9$, $P_m = 0.1$, $G_{max} = 300$ and $\varepsilon = 0.0001$. The parameters for HIG are as follows: $\eta = 4$, $\mathbf{T}_0 = 40$, $\lambda = 0.9$, $I_{fixed} = (n + m - 1) * 30$, $T_{final} = 1$ and $N_{max} = 25$.

B. Simulation Results

We evaluate the performance of hybrid algorithms and meta-heuristics under different experiment settings. We first investigate the impact of the number of mobile users on the performance of algorithms. The number of tasks is 80, 120 and 200 with different number of mobile users from 10 to 50. The results are shown in Fig.4, Fig.5 and Fig.6. We can deduce that the hybrid algorithms outperform the original meta-heuristic algorithms with lower penalty under different number of mobile users. This is because the hybrid algorithms utilize the ECT to generate a good initial solution, which reduces the blind search and enhances the optimization. Especially, the HGA has better performance than the HIG. For all the algorithms, the total penalty decreases as the number of mobile users increases. This is because the tasks can have more chance to be assigned to other mobile users with earlier completion time.

Second, we study the impact of the number of tasks on the performance of algorithms. We set the number of mobile users is 30 and vary the number of tasks from 40 to 120. As shown in Fig.7, the total penalty increases with the number of tasks. Since there are fixed number of mobile users, a mobile user will process more tasks on average, which may increase the completion time and the tardiness of tasks.

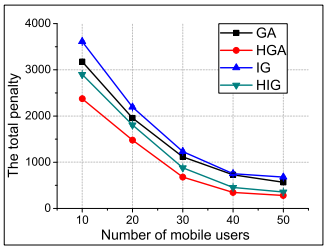

Fig. 4. The total penalty vs mobile users when the task number is 80

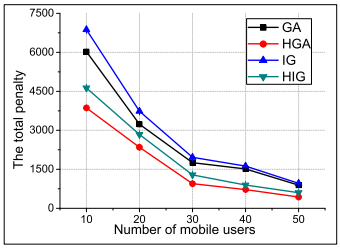

Fig. 5. The total penalty vs mobile users when the task number is 120

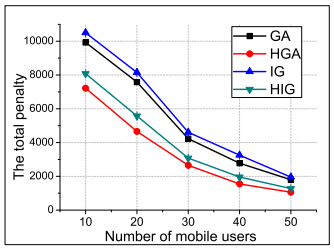

Fig. 6. The total penalty vs mobile users when the task number is 200

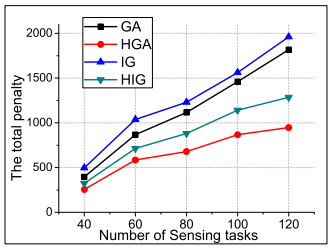

Fig. 7. The total penalty vs task number when the user number is 30

VI. Conclusion

In this paper, we have studied the heterogeneous task allocation problem in participatory sensing systems with the goal of minimizing the the total penalty caused by the tardiness of tasks. We formally formulate this problem and rigorously prove it is NP-hard. To address this problem, we have proposed two hybrid heuristic algorithms which combine the ECT heuristic with meta-heuristic algorithms. Extensive simulations have been performed to evaluate these algorithms. The results demonstrate that the hybrid algorithms outperform the meta-heuristic algorithms with lower total penalty. In the future work, we will investigate the situation where dynamic arrival and departure of users is allowed and the on-line model of tasks.

VII. Acknowledgement

This work was supported by the Natural Science Foundation of China (NSFC) projects (No. 61373155, 91438121, U1401253 and 61373156), China 863 Program (No. 2015AA01A202), the Key Basic Research Project (No. 12JC1405400), STCSM Project No. 13511507800 and the Shanghai Pujiang Program (No. 13PJ1404600) of the Shanghai Municipality.

References

[1] N.D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A.T. Campbell. A survey of mobile phone sensing. *Communications Magazine, IEEE*, 48(9):140–150, Sept 2010.

[2] R.K. Ganti, F. Ye, and H. Lei. Mobile crowdsensing: current state and future challenges. *Communications Magazine, IEEE*, 49(11):32–39, November 2011.

[3] E. Miluzzo, N.D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S.B. Eisenman, X. Zheng, and A.T. Campbell. Sensing meets mobile social networks: The design, implementation and evaluation of the cenceme application. In *Proc. ACM SenSys*, 2008.

[4] Z. Yu, Y. Feng, H. Xu, and X. Zhou. Recommending travel packages based on mobile crowdsourced data. *Communications Magazine, IEEE*, 52(8):56–62, 2014.

[5] J. Zhu, K. Zeng, K.-H. Kim, and P. Mohapatra. Improving crowd-sourced wi-fi localization systems using bluetooth beacons. In *Proc. IEEE SECON*, 2012.

[6] A. Rai, K.K. Chintalapudi, V.N. Padmanabhan, and R. Sen. Zee: Zero-effort crowdingsourcing for indoor localization. In *Proc. ACM SIGCOMM*, 2012.

[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/IEEE IPSN*, 2010.

[8] Y. Cheng, X. Li, Z. Li, S. Jiang, Y. Li, J. Jia, and X. Jiang. Aircloud: a cloud-based air-quality monitoring system for everyone. In *Proc. ACM SenSys*, 2014.

[9] P. Mohan, V.N. Padmanabhan, and R. Ramjee. Trafficsense: Rich monitoring of road and traffic conditions using mobile smartphones. Technical Report MSR-TR-2008-59, Microsoft Research, April 2008.

[10] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson. Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *Proc. ACM SenSys*, 2009.

[11] B. Guo, Z. Yu, X. Zhou, and D. Zhang. From participatory sensing to mobile crowd sensing. In *Proc. IEEE PERCOM Workshops*, 2014.

[12] Q. Zhao, Y. Zhu, H. Zhu, J. Cao, G. Xue, and B. Li. Fair energy-efficient sensing task allocation in participatory sensing with smartphones. In *Proc. IEEE INFOCOM*, 2014.

[13] S. He, D.-H. Shin, J. Zhang, and J. Chen. Toward optimal allocation of location dependent tasks in crowdsensing. In *Proc. IEEE INFOCOM*, 2014.

[14] D. Yang, G. Xue, X. Fang, and J. Tang. Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In *Mobicom*, 2012.

[15] 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.

[16] Z. Wang, D. Huang, H. Wu, Y. Deng, A. Aikebaier, and Y. Teranishi. Qos-constrained sensing task assignment for mobile crowd sensing. In *Proc. IEEE GLOBECOM*, 2014.

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

[18] I. Boutsis and V. Kalogeraki. On task assignment for real-time reliable crowdsourcing. In *ICDCS*, 2014.

[19] A. Allahverdi, C. Ng, T. E. Cheng, and M. Y. Kovalyov. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3):985–1032, 2008.

[20] Ü. Bilge, F. Kıraç, M. Kurtulan, and P. Pekgün. A tabu search algorithm for parallel machine total tardiness problem. *Computers & Operations Research*, 31(3):397–414, 2004.

[21] Da. E Golberg. Genetic algorithms in search, optimization, and machine learning. *Addion wesley*, 1989, 1989.

[22] C. Jou. A genetic algorithm with sub-indexed partitioning genes and its application to production scheduling of parallel machines. *Computers & Industrial Engineering*, 48(1):39–54, 2005.

[23] C. Liu. A hybrid genetic algorithm to minimize total tardiness for unrelated parallel machine scheduling with precedence constraints. *Mathematical Problems in Engineering*, 2013, 2013.

[24] R. Tavakkoli-Moghaddam, F Taheri, M Bazzazi, M Izadi, and F. Sassani. Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints. *Computers & Operations Research*, 36(12):3224–3230, 2009.

[25] L. W Jacobs and M. J Brusco. Note: A local-search heuristic for large set-covering problems. *Naval Research Logistics (NRL)*, 42(7):1129–1140, 1995.

[26] S.-W. Lin, Z.-J. Lee, K.-C. Ying, and C.-C. Lu. Minimization of maximum lateness on parallel machines with sequence-dependent setup times and job release dates. *Computers & Operations Research*, 38(5):809–815, 2011.