An Incentive Mechanism based on a Stackelberg Game for Mobile Crowdsensing Systems with Budget Constraint ⋆

Hamta Sedghani[a], Danilo Ardagna[b], Mauro Passacantando[c], Mina Zolfy Lighvan[a,*] and Hadi S. Aghdasi[a]

[a]Faculty of Electrical and Computer Engineering, University of Tabriz, Iran
[b]Department of Electronics, Information and Bioengineering, Polytechnic University of Milan, Milan, Italy
[c]Department of Computer Science, University of Pisa, Pisa, Italy

Keywords:
Mobile Crowdsensing
Incentive mechanism
Stackelberg game
Users' parameters estimation
Derivative-free optimization

Abstract

The adoption of smart device technologies is steadily increasing. Most of the smart devices in use today have built-in sensors which measure motion, direction, and various environmental conditions. Sensors are able to provide raw data with different quality and accuracy. A large group of smart devices forms a mobile crowdsensing system which is capable of sensing, collecting and sharing the environmental data to perform large scale sensing jobs. This paper aims to study and design an incentive mechanism for a mobile crowdsensing system based on a one-leader multi-follower Stackelberg game. A platform provider, as proponent of the sensing job, will act as the leader, while the mobile users will act as the followers. The final goal is to devise an efficient mechanism able to motivate the smart device users to participate in the sensing activity. Different from existing approaches, we propose a centralized method where the platform provider can estimate users' parameters very efficiently sending and receiving a few messages. We formulate the optimization problem on the platform provider side as a mixed integer nonlinear program with time constraints for each job and a budget constraint. Finally, a heuristic algorithm based on the derivative-free directional direct search method is designed to solve the platform optimization problem and achieve a close-to-optimal solution for the game. Results show that our Stackelberg game solution is much more scalable than the approach proposed in the work by other authors [1] as we can decrease the average number of messages by a factor between 53 to 80 and the average running time between 23 and 650 times. Furthermore, we compared our heuristic algorithm with BARON, a state of the art commercial tool for mixed integer global optimization, to solve the platform optimization problem. Results demonstrated that our proposed algorithm converges to a near-optimal solution much faster especially in large scale systems.

1. Introduction

Mobile crowdsensing (MCS) is a newly emerging paradigm which uses smartphone technologies and capabilities for environmental data collection and analysis. However, the implementation of advanced environmental monitoring applications can be costly and complex [2]. For example, users equipped with portable devices are able to perform jobs like measuring the noise level by microphones or quantifying the environment visibility by built-in cameras [3, 4, 5], map semantic labeling [6, 7, 8, 9, 10], perform road condition monitoring [11, 12, 13], estimate pollution [14, 15] and traffic [16], or they can collect crowd-source images [17, 18] performing, e.g., target identification [19, 20, 21, 22, 23]. A new wave of applications is expected from the adoption of augmented reality sensors, e.g., LIDAR available in new generation phones and tablets [24].

Smartphone users, by using the embedded sensors in their devices, act as "participants" to sense environmental data and share these data using the existing communication infrastructure [25]. Therefore, MCS could be an appropriate alternative to the traditional wireless sensor networks for sensing, monitoring and coverage [26, 27, 28].

Generally, a mobile crowdsensing system consists of two parts. The first part is the crowdsensing platform which initiates many mobile crowdsensing jobs and provides some related services, such as pushing crowdsensing tasks to users' smartphones. The second part includes the set of users' smartphones which provide sensing data according to the crowdsensing task requirements published by the platform. The most important challenge in MCS is that users have to be encouraged to participate in the MCS system. To achieve this goal, most existing works use an incentive mechanism to motivate the users to participate. These incentive mechanisms only need to guarantee rationality, i.e., the income of smartphone users will be more than their cost for sensing. Several existing works use auction [29, 30, 31, 32] as an incentive mechanism, while some others rely on game theory [1, 33, 34, 35, 36].

Most of game-theory-based literature approaches consider a quadratic function [1, 35, 36, 37, 38] as individual user's cost function and use a distributed algorithm to find an equilibrium instead of solving the game in a central manner. This is intuitive, since the user's cost function parameters are private and unknown to the platform. With respect to a centralized approach, a distributed mechanism requires more

⋆This paper has been published in Ad Hoc Networks, vol. 123, Article 102626, see: https://www.sciencedirect.com/science/article/abs/pii/S1570870521001505.
*Corresponding author
h.sedghani@tabrizu.ac.ir (H. Sedghani); danilo.ardagna@polimi.it (D. Ardagna); mauro.passacantando@unipi.it (M. Passacantando); mzolfy@tabrizu.ac.ir (M.Z. Lighvan); aghdasi@tabrizu.ac.ir (H.S. Aghdasi)
ORCID(s):

time and resources especially if the number of users and iterations are large. Indeed, at each iteration the platform should communicate with all users to determine the price and time allocation for the sensing task while, in a central manner, most of this communication can be eliminated saving data transfer and computation time. Moreover, many works' goal is the social welfare maximization [1, 35, 36, 39, 40, 41], but the social welfare cannot be considered a good measurement in a realistic situation because in practice a sensing process is initiated by a platform which leads to a non-cooperative model. Platform and users are going to maximize their profit and minimize their costs (e.g., minimizing the total cost for a sensing job is one of the platform goals while minimizing the battery consumption is one of the users' goal). Therefore, from a technical point of view, finding the equilibrium in a non-cooperative game is much more difficult and also more realistic and important than maximizing the social welfare.

In this paper, we consider a platform issuing multiple sensing jobs and a large number of mobile users with smart devices randomly distributed in a wide area. We assume that the platform knows the sensing quality of the mobile users thanks to the users' sensing information history [1, 42]. The platform has a budget constraint for all jobs and it also defines lower and upper limits for the execution time and cost of each job (indeed, a job needs a minimum time to be considered trusted and accurate and has a maximum time to avoid additional costs). Users also define time constraints for their participation because their devices have some time-dependent restrictions, such as battery consumption.

This paper aims to design an incentive mechanism that motivates users to participate while maximizes the platform's utility. Contrary to other works, in this paper we consider a model based on a one-leader multi-follower Stackelberg game, where the platform estimates the users' cost function parameters by relying on Karush-Kuhn-Tucker (KKT) conditions. Then, the game is solved in a central manner on the platform side, saving computational time and resources.

We extend a previous work by other authors [1], which proposed a distributed approach based on a dual decomposition method to maximize the social welfare of the system. The Stackelberg game is more realistic in practice and more challenging from a technical perspective with respect to [1]. We solve the optimization problem from the users' side by solving the corresponding KKT system in closed form. The KKT solution is used to develop a simple algorithm to estimate the users' parameters. In this way, the platform can solve its optimization problem in a central manner instead of a distributed one. Moreover, we use the users' KKT systems to reformulate the Stackelberg game as a mixed integer nonlinear program. Finally, we propose a derivative-free heuristic algorithm which runs on the platform side and we compare our centralized approach with the distributed approach developed in [1]. We show that our solution is faster and more cost effective even neglecting the network time, since the number of messages exchanged among the platform and its users is significantly reduced. We also compare the proposed heuristic algorithm with the commercial solver BARON [43] and we demonstrate that our algorithm outperforms BARON especially for large scale systems, since it converges much faster than BARON while achieving almost the same solution.

In summary, the main contributions of this paper are the following:

1. We formulate the interaction among multiple mobile users and a platform as a one-leader multi-follower Stackelberg game, where the platform is the leader and the users are the followers.

2. We develop a method which allows the platform to estimate the parameters of the users' cost functions by sending a few messages containing the prices and receiving users time dedication as responses. We also propose a derivative-free heuristic method for solving the game on the platform side.

3. We finally evaluate the performance of the proposed approach through extensive numerical experiments providing also a comparison with another literature proposal [1]. Experimental results show that our heuristic algorithm can converge to the optimal solution quickly. With respect to the work in [1], our approach obtains at least 53x network saving and at least 23x computational time reduction (event neglecting message transfer time). Moreover, we also obtain a platform net utility increase between 0.6% and 20.5%.

The remainder of this paper is organized as follows. Related works are discussed in Section 2. Section 3 describes the MCS system model and formulates the problem as a Stackelberg game. In Section 4, we illustrate the algorithm that allows the platform to estimate the users' parameters. In Section 5, we reformulate the Stackelberg game as a mixed integer nonlinear program, while Section 6 presents our heuristic approach to solve the Stackelberg game. Experimental results are discussed in Section 7, while conclusions are finally drawn in Section 8.

2. Related work

With the development of smart devices and wireless networks in people's lives, mobile crowdsensing has become an extensive research area [44]. Incentive mechanism design is one of the most important issues in order to attract mobile users to participate in MCS systems and common methods to incentivize mobile users include auction, reverse auction, game theory, etc.

Feng et al. [39] designed a framework based on reverse auctions called TRAC for modeling the interactions among the platform and the mobile users in which mobile users are the sellers and the platform is the buyer (buying sensing services).

In [45, 46] authors considered two system models, a crowdsourcer-centric model and a user-centric model, without considering, however, a budget constraint for the crowdsourcer. In the crowdsourcer-centric model, the authors

proposed a Stackelberg game-based incentive mechanism, where the crowdsourcer is the leader while the users are the followers. In the first step, the crowdsourcer declares its reward R, therefore R is the strategy of crowdsourcer. In the second step, users announce their sensing time which is their strategy to maximize their own utility. Authors proved that the game has a unique equilibrium and designed a mechanism for computing it. In this way, the crowdsourcer is able to maximize its utility while all users are playing their best response strategy. In the user-centric model, authors proposed an incentive mechanism based on auction, which is profitable, truthful, computationally efficient and individually rational.

Luo et al. [47] designed an incentive mechanism based on asymmetric auction for heterogeneous crowdsourcing, which accommodates an arbitrary number of heterogeneous users with incomplete information. They demonstrated that the mechanism persuades the self-interested users to make their best effort while minimizing the cost of the crowdsourcer, and outperforms traditional mechanisms that consider a fixed price in both symmetric and asymmetric auction.

There are many works based on game theory to design an incentive mechanism for MCS. Wang et al. [33] have proposed a two-level pricing scheme which is time-sensitive and location-dependent with random users arrivals to balance the participation of users among tasks. At the first level, the reward for each task is different from others and it depends on popularity of tasks which can be obtained from spatio-temporal inequality of tasks. At the second level, the reward of each task to each user will dynamically change when the demand of a user changes. Moreover, authors proved that the task allocation problem for randomly arriving users with time budget is NP-hard, and proposed some greedy algorithms to solve such a problem.

Peng et al. [34] have designed a bilateral competition framework where crowdsourcers compete for the limited sensing service and smart devices compete for the limited budget of the crowdsourcers. Each crowdsourcer has to select an "optimal" budget that can attract enough smart devices to participate. Each smart device participator has to decide the crowdsourcer to join.

Zhan et al. [35] designed an incentive mechanism and analyzed the interaction among the crowdsensing platform and the smart device users relying on Nash bargaining theory for the platform-centric mobile crowdsensing. They also designed a distributed algorithm based on dual decomposition method which can keep the participators' privacy and reduce the sensing-platform's computation load.

Nie et al. [48] have proposed an incentive model based on two-stage Stackelberg game and achieved the equilibrium by backward induction. They considered the social network effects among users and, based on that, they have developed discriminatory incentive and uniform incentive mechanisms, obtaining the closed-form expression for the optimal incentive. Furthermore, they have analyzed the interaction between the platform and mobile users where the social structure information, i.e., the social network effects, is uncertain and formulated as a Bayesian Stackelberg game with incomplete information.

Duan et al. [36] designed a distributed algorithms to compute the Walrasian equilibrium and they used a dual decomposition method to solve the social welfare maximization problem. They assumed that the platform pays the same price to all the participators in the same sensing area.

Conversely, the authors in [1] considered different prices for different quality of the sensed data contributed by individual users due to the influence of various factors (e.g., sensor quality, noise, etc.). They considered multiple platforms but they assumed that each platform has a single job. Authors also assumed that the sensing quality of mobile users is known by the sensing platforms, while the cost functions of the mobile users and utility parameters for the sensing platforms are their own private information. Therefore, they proposed to maximize the social welfare of the system, since the private nature of the mobile users makes it impossible to solve the problem in a central manner. Then, they developed a distributed iterative method based on dual decomposition to divide the social welfare maximization problem into sensing platforms' local optimization problems and mobile users' local optimization problems. The distributed algorithm, based on an iterative gradient descent method, is designed to achieve the close-to-optimal solution. Since the method requires to solve an optimization problem in both platform and users side for each iteration, it is very time consuming and also needs to exchange many messages (that leads to significant resource consumption).

Similarly to [1], in this work we consider a sensing area including many users where the platform will issue the sensing jobs. The mobile users are characterized by different sensing quality devices and the sensing platform will consider these different qualities when it employs them to participate in the mobile crowdsensing application. Contrary to previous literature proposals, we do not maximize the social welfare of the whole system because the platform can estimate the private parameters of the users and therefore we do not have to use a distributed method. After the parameters estimation, we apply a derivative-free direct search algorithm to maximize the platform net profits.

3. Problem formulation

In this section, we introduce the MCS system model (Section 3.1) and formulate the MCS problem as a Stackelberg game (Section 3.2) detailing both the platform and users underlying optimization problems.

3.1. Basic settings

We consider a MCS system consisting of a platform P and a set $U = \{u_1, u_2, \dots, u_N\}$ of users with smart devices (smartphones, tablets, etc.) connected to the platform via Internet. The sensing platform wants to incentivize the users to participate in completing its sensing jobs set indexed by $J = \{1, 2, \dots, K\}$ by paying some money. We suppose each user u_i selects a subset $J_i \subseteq J$ of jobs she/he will perform and spends t_{ik} time units for completing job k.

Different jobs have different prices per time unit due to their intrinsic characteristics such as complexity and cost. For example, users expect more returns when performing jobs that require graphic or video information than those that only require audio information [1]. Besides, a job can be performed in a specific location with different prices for different users (due, e.g., to the quality of information provided by users and/or users' regional differences, and/or by their relative importance). Therefore, the quality of user not only depends on his/her smart device, but also the location of the job execution. For example, if we are interested in sensing data for noise pollution monitoring, then data from a noisy region is more important than data from relatively noiseless areas. Indeed, in a noiseless region few samples might be sufficient for the analysis of interest, while in a noisy environment, which might be characterized, e.g., also by some variability in the noise level, higher quality sensing might be relevant. So, for the aforementioned reasons, we consider different prices for different jobs and users and we denote with p_{ik} the price per time unit that the platform pays to user u_i for job k.

Remark 1. Each job considered in this paper will be executed on one specific location with a specified coverage radius. Similar to other literature proposals [1, 36, 40, 41, 45], we assume that all mobile users, who received the platform payment, will execute the jobs. In other words, all volunteers who want to participate in MCS do not move from the coverage area while performing their task. Vice versa, if the mobility matters, we assume that the users do not move during all the phases of our framework including: estimating the parameters of users, running the heuristic algorithm to obtain the optimal times and prices, and executing the sensing jobs. Since, as will be discussed in the experimental section, our proposed approach converges quickly (for example, for a 1000 users scenario, the total time required by our approach is less than 52 seconds), we argue that such assumptions are reasonable in several scenarios of practical interest.

3.2. A Stackelberg game model

In the system under study, the platform has to determine the optimal prices p_{ik} in order to maximize its own net utility, in such a way all jobs are completed and a budget constraint is fulfilled. On the other hand, each user has to find the optimal time units t_{ik} to maximize her/his own profit without violating her/his resource constraint. Since the platform and users are selfish and the users choices follow the choices of the platform, we propose a one-leader multi-follower Stackelberg game approach [49] to model the MSC system, where the platform acts as leader and the users as followers. Figure 1 shows this relation. The detailed optimization problems solved by the users and the platform are described in the following.

User profit model. When mobile user u_i allocates t_{ik} time units to job k, she/he acquires $p_{ik}t_{ik}$ amount of money from the platform and incurs a cost for completing the job. The cost function $C_{ik}(t_{ik})$ for job k incorporates several factors

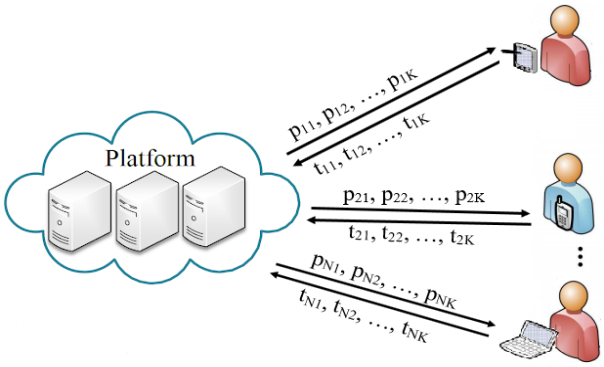


Figure 1: Interactions among the platform and its users.

such as physical or mental tiredness of mobile users, battery drainage, and bandwidth occupation of mobile devices, etc. We assume that C_{ik} is a strongly convex quadratic function as in many other literature proposals (see, e.g., [1, 36, 37, 38, 50, 51, 52, 53]):

$$C_{ik}(t_{ik}) = \frac{1}{2}a_{ik}t_{ik}^2 + b_{ik}t_{ik} + c_{ik}, \tag{1}$$

where $a_{ik}, b_{ik} > 0$ are parameters depending on both user availability level and job difficulty level [36] and $c_{ik} \geq 0$. Moreover, each user u_i has an upper bound T_i in her/his sensing time due to her/his resource constraints. Given the prices p_{ik} chosen by the platform, user u_i has to determine the vector $\mathbf{t}_i = (t_{ik})_{k\in J_i}$ by solving the following optimization problem:

$$\max_{\mathbf{t}_i} \sum_{k\in J_i} \left[p_{ik}\, t_{ik} - C_{ik}(t_{ik})\right] \tag{2}$$

subject to:

$$\sum_{k\in J_i} t_{ik} \leq T_i, \tag{3}$$

$$t_{ik} \geq 0 \qquad \forall\, k \in J_i. \tag{4}$$

Notice that, for any given prices p_{ik}, the problem (2)–(4) has a unique optimal solution, since the objective function is strongly convex and the feasible region is a bounded polyhedron.

Platform profit model. On the platform side, we assume that the platform provider knows the sensing quality of the mobile users due to the history of the sensing information. There are many methods to evaluate the mobile users' quality (see, e.g., [42, 54, 55]). We assume that ω_{ik} is the sensing quality of user u_i relative to the job k, for any $k \in J_i$. Moreover, the platform has a utility function ϕ_k, for each job $k \in J$, that depends on the vector

$$\mathbf{t} = (t_{ik})_{i=1,\ldots,N,\, k\in J_i}$$

of time variables controlled by the users and the sensing quality parameters ω_{ik}. We define the utility function ϕ_k as

follows:

$$\phi_k(\mathbf{t}) = \mu_k \log\left(1 + \sum_{i=1}^{N} \log(1 + t_{ik}\,\omega_{ik})\right), \tag{5}$$

where μ_k is a system parameter, the $\log(1 + t_{ik}\,\omega_{ik})$ term reflects the diminishing marginal utility gain on the sensing contribution of user u_i for job k, and the outer log term reflects the diminishing marginal utility gain on the number of participating users. Such a function has been widely used in previous works for MCS systems (see, e.g., [35, 36, 40, 45, 46, 56]). Notice that ϕ_k satisfies the following two conditions [45]:

1. ϕ_k increases with the users' sensing time contributions, i.e., $\dfrac{\partial \phi_k}{\partial t_{ik}} > 0$ holds for any $i = 1, \dots, N$;
2. the growth rate of ϕ_k decreases with the users' sensing time increasing, i.e., $\dfrac{\partial^2 \phi_k}{\partial t_{ik}^2} < 0$ holds for any $i = 1, \dots, N$, which is also known as diminishing returns in economics [57].

The platform defines a minimum and a maximum time unit cost P_k^L and P_k^U for any job k that it is willing to pay for that job. Moreover, the platform has a given budget B to reward users for their services. Finally, the total time spent by all the users for job k has both a lower bound t_k^L, since the platform has to collect enough sensing data to guarantee the overall performance, and an upper bound t_k^U, when no more data are needed to accomplish the sensing task in order to avoid extra costs.

The platform has to determine the prices p_{ik}, for any $i = 1, \dots, N$ and $k \in J_i$ (we denote $\mathbf{p} = (p_{ik})_{i=1,\dots,N,\,k\in J_i}$), by solving the following bilevel optimization problem:

$$\max_{\mathbf{p}} \; F(\mathbf{p}) := \sum_{k\in J} \phi_k(\mathbf{t}^*(\mathbf{p})) - \sum_{i=1}^{N} \sum_{k\in J_i} p_{ik}\, t_{ik}^*(\mathbf{p}) \tag{6}$$

subject to:

$$\mathbf{t}_i^*(\mathbf{p}) = \arg\max_{\mathbf{t}_i} \left\{ \sum_{k\in J_i} \left[p_{ik}\, t_{ik} - C_{ik}(t_{ik})\right] \; : \; \sum_{k\in J_i} t_{ik} \le T_i,\; t_{ik} \ge 0,\; \forall\, k \in J_i \right\} \quad \forall\, i = 1, \dots, N, \tag{7}$$

$$P_k^L \le p_{ik} \le P_k^U \qquad \forall\, i = 1, \dots, N,\; \forall\, k \in J_i, \tag{8}$$

$$\sum_{i=1}^{N} \sum_{k\in J_i} p_{ik}\, t_{ik}^*(\mathbf{p}) \le B, \tag{9}$$

$$t_k^L \le \sum_{i:\; k\in J_i} t_{ik}^*(\mathbf{p}) \le t_k^U \qquad \forall\, k \in J. \tag{10}$$

The objective function (6) represents the net utility of the platform and is equal to the difference between the sum of utility functions and the total payment received by users.

Table 1
Parameters and Decision Variables

	Parameters
N	Number of users
K	Number of platform's jobs
J	Set of platform's jobs
J_i	Set of jobs selected by user u_i
T_i	Maximum time budget of user u_i
B	Maximum platform budget
P_k^L	Minimum time unit cost for job k set by users
P_k^U	Maximum time unit cost for job k set by the platform
t_k^L	Lower bound of the total time spent for job k
t_k^U	Upper bound of the total time spent for job k
	Decision Variables
p_{ik}	Price paid by the platform to user u_i for job k
t_{ik}	Time spending of user u_i for job k

Constraints (7) guarantee that $t_{ik}^*(\mathbf{p})$ are the optimal times found by the users as best response to the prices p_{ik} chosen by the platform. Constraints (8) impose that each price p_{ik} satisfies the lower and upper bounds defined by the platform. Constraint (9) entails that the total payment received by users is lower than the maximum budget of the platform. Finally, constraints (10) guarantee that the total time devoted by users to each job is between the minimum and the maximum time requirements for the job.

Summarizing, the problem (6)–(10) represents the one-leader multi-follower Stackelberg game model, where the platform (leader) controls the price vector $\mathbf{p}$, while the users (followers) control the time vectors $\mathbf{t}_i$, for any $i = 1, \dots, N$. For the sake of clarity, the notation used in this paper is summarized in Table 1.

4. Estimation of users' parameters

In order to find a solution of the Stackelberg game, the platform needs to anticipate the best response of each user to any price $\mathbf{p}$. To this end, it has to estimate the *a priori unknown* parameters a_{ik} and b_{ik} appearing in the cost function, and the time budget T_i for each user u_i. In this section, first we show a general formula to compute the optimal solution of each user's problem (Section 4.1), then we develop an algorithm based on the latter formula to estimate the user parameters a_{ik}, b_{ik} and T_i (Section 4.2).

Remark 2. As in other literature proposals [1, 36], we assume that the users are truthful when declaring their time willingness t_{ik} to participate to a job or setting the upper time limit T_i, while they would like not to share their private parameters a_{ik}, b_{ik}. Note that, this paper main contributions are: (i) an extension of previous work [1] with the development of a solution to estimate the private users' parameters, and (ii) a novel centralised and efficient method for solving the general problem (6)–(10).

4.1. Optimal solution of the user's problem

Let us consider the user u_i's problem. For any job $k \in J_i$, the cost C_{ik} is a strongly convex quadratic function of t_{ik}, i.e., $C_{ik}(t_{ik}) = \frac{1}{2}a_{ik}t_{ik}^2 + b_{ik}t_{ik} + c_{ik}$, where $a_{ik}, b_{ik} > 0$ and $c_{ik} \geq 0$. User u_i has to solve the following problem:

$$\max_{\mathbf{t}_i} \sum_{k \in J_i} \left[-\frac{1}{2}a_{ik}t_{ik}^2 + (p_{ik} - b_{ik})t_{ik} - c_{ik} \right] \tag{11}$$

subject to:

$$\sum_{k \in J_i} t_{ik} \leq T_i, \tag{12}$$

$$t_{ik} \geq 0 \qquad \forall\, k \in J_i. \tag{13}$$

We denote $d_{ik} := p_{ik} - b_{ik}$ for any $k \in J_i$, and define the subsets of jobs

$$J_i^- = \left\{ k \in J_i : \; d_{ik} \leq 0 \right\}, \quad J_i^+ = \left\{ k \in J_i : \; d_{ik} > 0 \right\}.$$

Without loss of generality, we can assume that $J_i^+ = \{1, \dots, n\}$ and $d_{i1} \leq d_{i2} \leq \cdots \leq d_{1n}$.

Theorem 4.1. *The optimal solution $\mathbf{t}_i^* = (t_{ik}^*)_{k \in J_i}$ of problem (11)–(13) is given as follows:*

a) $t_{ik}^* = 0$ for any $k \in J_i^-$.

b) To find t_{ik}^* for $k \in J_i^+$, we distinguish $n+1$ cases:

- *If $\sum_{k=1}^{n} \frac{d_{ik}}{a_{ik}} \leq T_i$, then $t_{ik}^* = \frac{d_{ik}}{a_{ik}}$ for any $k = 1, \dots, n$.*
- *If $z_1 := \dfrac{\sum_{k=1}^{n} \frac{d_{ik}}{a_{ik}} - T_i}{\sum_{k=1}^{n} \frac{1}{a_{ik}}} \in (0, d_{i1})$, then*

  $t_{ik}^* = \dfrac{d_{ik} - z_1}{a_{ik}}$ *for any $k = 1, \dots, n$.*
- *Let $q \in \{2, \dots, n\}$.*

  If $z_q := \dfrac{\sum_{k=q}^{n} \frac{d_{ik}}{a_{ik}} - T_i}{\sum_{k=q}^{n} \frac{1}{a_{ik}}} \in [d_{i,q-1}, d_{iq})$, then

$$t_{ik}^* = \begin{cases} 0 & \text{if } k = 1, \dots, q-1, \\ \dfrac{d_{ik} - z_q}{a_{ik}} & \text{if } k = q, \dots, n. \end{cases}$$

Proof. The proof relies on the solution of the Karush-Kuhn-Tucker system of the user's problem and is given in Appendix A. □

The formulas shown in Theorem 4.1 have the following interpretation. If the price p_{ik} offered by the platform for job k is below the threshold b_{ik}, then the optimal time to devote to the job k is $t_{ik}^* = 0$. If $p_{ik} > b_{ik}$, then its value depends on the time budget T_i: if T_i is large enough (i.e., greater than $\sum_{k=1}^{n}(p_{ik} - b_{ik})/a_{ik})$ then each t_{ik}^* is equal to the ratio $(p_{ik} - b_{ik})/a_{ik}$; otherwise the optimal times have to be reduced according to the ratios $z_1, \dots, z_n$.

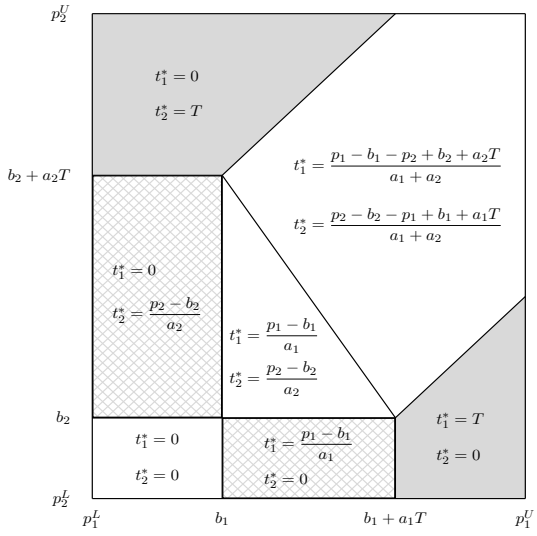


Figure 2: Optimal time for the two jobs scenario.

Notice that the solution given in Theorem 4.1 is general for any number of jobs. In particular, if a user selects only one job, then its optimal time is given by the following simple formula (where index i has been omitted):

$$t_1^* = \begin{cases} 0 & \text{if } p_1 \in [P_1^L,\ b_1], \\ \dfrac{p_1 - b_1}{a_1} & \text{if } p_1 \in [b_1,\ b_1 + a_1 T], \\ T & \text{if } p_1 \in [b_1 + a_1 T,\ P_1^U]. \end{cases} \tag{14}$$

A graphical description of the optimal time for the two jobs scenario is shown in Figure 2, while the analytical optimal solution for the three jobs scenario is given in Appendix B.

In the next section, we will show how formula (14) can be exploited to estimate parameters a, b and T.

4.2. Parameters estimation

The parameters a, b and T of all users can be estimated by the platform through Algorithm 1. The algorithm performs the following steps for each user u_i.

First, the platform estimates the time budget T_i (lines 3-8). It chooses a job $h \in J_i$ and sets a very large number as a price for job h, while it sets the minimum time unit cost P_k^L for other jobs. Then, the platform sends the prices to user u_i and receives the user's optimal times (lines 3-7). According to Theorem 4.1 (see also Figure 2), if the time unit price of a job is very high while the ones for other jobs are minimum, then the optimal time for job h will be equal to user's maximum time budget T_i (line 8).

Then, the platform estimates the parameters a_{ik} and b_{ik} for each job k (lines 9-27). Intuitively, the platform will generate a sequence of prices to induce the user's best reply trajectory

Algorithm 1 Estimation of users' parameters

1: Choose $\varepsilon > 0$ small enough and $M > 0$ large enough
2: **for** $i = 1, \dots, N$ **do**
3: Choose a job $h \in J_i$
4: $p_{ih} \leftarrow M\, P_h^U$
5: $p_{ij} \leftarrow P_j^L$ for any $j \in J_i \setminus \{h\}$
6: The platform sends the prices to user u_i
7: User u_i sends her/his optimal solution t_i to the platform
8: $T_i \leftarrow t_{ih}$
9: **for** $k \in J_i$ **do**
10: $p_{ij} \leftarrow P_j^L \quad \forall j \in J_i \setminus \{k\}$
11: **repeat**
12: The platform randomly picks $p'_{ik} \in [P_k^L, P_k^U]$
13: $p_{ik} \leftarrow p'_{ik}$
14: The platform sends the prices to user u_i
15: User u_i sends her/his optimal solution t'_i to the platform
16: **until** $0 < t'_{ik} < T_i$
17: $p''_{ik} \leftarrow p'_{ik} + \varepsilon$
18: The platform sends p''_{ik} to user u_i
19: User u_i sends her/his optimal solution t''_{ik} to the platform
20: **if** $t''_{ik} = T$ **then**
21: $p''_{ik} \leftarrow p'_{ik} - \varepsilon$
22: The platform sends p''_{ik} to user u_i
23: User u_i sends her/his optimal solution t''_{ik} to the platform
24: **end if**
25: $a_{ik} \leftarrow \dfrac{p''_{ik} - p'_{ik}}{t''_{ik} - t'_{ik}}$
26: $b_{ik} \leftarrow p'_{ik} - t'_{ik} \dfrac{p''_{ik} - p'_{ik}}{t''_{ik} - t'_{ik}}$
27: **end for**
28: **end for**

(t^*_{ik}) to a region where the relations $t^*_{ik} = (p_{ik} - b_{ik})/a_{ik}$ and $t^*_{ij} = 0$, for any $j \neq k$, hold (e.g., in the case of two jobs, the user best time distribution lays in squared rectangles in Figure 2). The platform sets the minimum price P_j^L for jobs $j \neq k$, so that the optimal time $t^*_{ij} = 0$ for any $j \neq k$. Moreover, it chooses a random price p'_{ik} in interval $[P_k^L, P_k^U]$ for job k. It sends the chosen prices to user u_i until the optimal time t'_{ik} for job k is greater than zero and strictly lower than T_i (lines 10-16). Formula (14) guarantees that a_{ik} and b_{ik} are related through the following relation:

$$t'_{ik} = \frac{p'_{ik} - b_{ik}}{a_{ik}}. \tag{15}$$

Then, the platform increases the price p'_{ik} by a small number ε (line 17) to obtain a new optimal time $t''_{ik} \in (0, T_i)$, so that a relation similar to (15) holds. The platform sends the new price p''_{ik} to u_i (line 18). Note that, if the the user replies with $t''_{ik} = T_i$, then the platform will decrease p'_{ik} by ε to receive an optimal time different from T_i (lines 21-22, this scenario corresponds to the gray areas in Figure 2 for the two jobs scenario). Finally, the platform can compute the parameters a_{ik} and b_{ik} (lines 25-26) by solving the following linear system in the variables a_{ik} and b_{ik}:

$$\begin{cases} t'_{ik} = \dfrac{p'_{ik} - b_{ik}}{a_{ik}}, \\ t''_{ik} = \dfrac{p''_{ik} - b_{ik}}{a_{ik}}. \end{cases} \tag{16}$$

5. A solution approach based on a single-level optimization reformulation

In this section, we show that the Stackelberg game (6)–(10) can be equivalently reformulated as a Mixed-Integer Nonlinear optimization problem.

First, we notice that, for each $i = 1, \dots, N$, the vector $t_i^*(\mathbf{p})$ is well-defined for any price vector $\mathbf{p}$, since the max problem in (7) has a unique optimal solution. Moreover, exploiting the Karush-Kuhn-Tucker (KKT) optimality conditions, constraint (7) is equivalent to the following system of equalities and inequalities:

$$\begin{aligned}
& 2a_{ik}t^*_{ik} + b_{ik} - p_{ik} + \lambda_i \geq 0 && \forall\, k \in J_i, \\
& t^*_{ik} \geq 0 && \forall\, k \in J_i, \\
& t^*_{ik}\left[2a_{ik}t^*_{ik} + b_{ik} - p_{ik} + \lambda_i\right] = 0 && \forall\, k \in J_i, \\
& \lambda_i \geq 0, \\
& \sum_{k \in J_i} t^*_{ik} \leq T_i, \\
& \lambda_i \left(T_i - \sum_{k \in J_i} t^*_{ik} \right) = 0,
\end{aligned}$$

where λ_i is the KKT multiplier associated to the user u_i's time budget constraint.

The latter nonlinear system is in turn equivalent to the following system containing binary variables:

$$\begin{aligned}
& 0 \leq t^*_{ik} \leq M_{ik}\, y_{ik} && \forall\, k \in J_i, \\
& 0 \leq 2a_{ik}t^*_{ik} + b_{ik} - p_{ik} + \lambda_i \leq M_{ik}\,(1 - y_{ik}) && \forall\, k \in J_i, \\
& 0 \leq \lambda_i \leq M'_i\, x_i, \\
& 0 \leq T_i - \textstyle\sum_{k \in J_i} t^*_{ik} \leq M'_i\,(1 - x_i), \\
& y_{ik} \in \{0, 1\} && \forall\, k \in J_i, \\
& x_i \in \{0, 1\},
\end{aligned}$$

where the constants M_{ik} and M'_i have to be chosen sufficiently large, e.g.,

$$M_{ik} \geq \max\left\{T_i,\ 2a_{ik}T_i + b_{ik} - p_k^U + \lambda_i\right\}$$

and $M'_i \geq \max\left\{\lambda_i,\ T_i\right\}$.

Therefore, the Stackelberg game (6)–(10) is equivalent to the following Mixed-Integer Nonlinear optimization problem:

$$\max_{(\mathbf{p},\mathbf{t},\lambda,\mathbf{x},\mathbf{y})} \sum_{k=1}^{K} \phi_k(\mathbf{t}) - \sum_{i=1}^{N} \sum_{k \in J_i} p_{ik}\, t_{ik} \tag{17}$$

subject to:

$$0 \le t_{ik} \le M_{ik}\, y_{ik} \qquad \forall\, i,\ \forall\, k \in J_i \qquad (18)$$

$$0 \le 2a_{ik}t_{ik} + b_{ik} - p_{ik} + \lambda_i \qquad \forall\, i,\ \forall\, k \in J_i \qquad (19)$$

$$2a_{ik}t_{ik} + b_{ik} - p_{ik} + \lambda_i \le M_{ik}\,(1 - y_{ik}) \qquad \forall\, i,\ \forall\, k \in J_i \qquad (20)$$

$$0 \le \lambda_i \le M_i'\, x_i \qquad \forall\, i \qquad (21)$$

$$0 \le T_i - \sum_{k \in J_i} t_{ik} \le M_i'\,(1 - x_i) \qquad \forall\, i \qquad (22)$$

$$\sum_{i=1}^{N} \sum_{k \in J_i} p_{ik}\, t_{ik} \le B \qquad (23)$$

$$t_k^L \le \sum_{i:\ k \in J_i} t_{ik} \le t_k^U \qquad \forall\, k = 1, \dots, K \qquad (24)$$

$$P_k^L \le p_{ik} \le P_k^U \qquad \forall\, i,\ \forall\, k \in J_i \qquad (25)$$

$$\lambda_i \ge 0 \qquad \forall\, i \qquad (26)$$

$$x_i \in \{0, 1\} \qquad \forall\, i \qquad (27)$$

$$y_{ik} \in \{0, 1\} \qquad \forall\, i,\ \forall\, k \in J_i. \qquad (28)$$

From a theoretical point of view, since the above Mixed-Integer Nonlinear program is equivalent to the Stackelberg game (6)–(10), the platform can use any commercial global optimization tool for its solution. However, a global optimal solution of the problem (17)–(28) is computationally very hard to find in practice, since the objective function (17) is not concave, constraint (23) is not convex, and the variables $\mathbf{x}$ and $\mathbf{y}$ are binary. In the next section, we propose a heuristic algorithm to find an approximate solution of the Stackelberg game (6)–(10). Numerical experiments in Section 7 will show that the proposed heuristic algorithm is computationally more efficient than the state-of-the-art global solver BARON [43].

6. A heuristic solution approach

In this section, we describe a heuristic algorithm (see Algorithm 2) for the approximate solution of the Stackelberg game (6)–(10).

First, Algorithm 2 estimates the users' parameters (line 1) by applying Algorithm 1 discussed in Section 4. This step will allow the platform to compute the users' optimal times $t^*(\mathbf{p})$ for any price vector $\mathbf{p}$ by Theorem 4.1.

Next, an initial feasible price vector $\mathbf{p}$ is found (line 6) by means of the Initialization function (see Algorithm 3). The Initialization function tries to increase the price of each job to incentivise users to participate as far as the constraints (8), (9) and (10) are not violated. If the optimal time vector $\mathbf{t}^*$ does not satisfy the time constraint (10) for a job k, then the platform increases p_{ik} by δ until the time constraints will be satisfied (lines 6-11). On the other hand, if the budget constraint (9) is not satisfied, the platform decreases any price p_{ik} by δ until the budget constraint will be satisfied (lines 13-18). Finally, the Initialization function computes two lists ΔF^+ and ΔF^- containing the incremental ratios of the platform net utility

Algorithm 2 Heuristic algorithm

1: Estimate the users' parameters by using Algorithm 1
2: $\alpha \leftarrow random(0, 0.1)$
3: $\beta_1 \leftarrow random(0, 0.5),\ \beta_2 \leftarrow random(\beta_1, 1)$
4: $\gamma \leftarrow random(1, 2)$
5: $\delta \leftarrow \alpha \min_{k \in J}\{P_k^U - P_k^L\}$
6: $(\mathbf{p}, \Delta F^+, \Delta F^-) \leftarrow$ INITIALIZATION(δ)
7: Compute platform **platform net utility** $F(\mathbf{p})$ by (6)
8: $CurrentF \leftarrow F(\mathbf{p})$
9: $z \leftarrow 0,\ flag \leftarrow False$
10: **while** ($z < MaxIteration$) AND ($flag = False$) **do**
11: $\quad \delta \leftarrow \alpha \min_{k \in J}\{P_k^U - P_k^L\}$
12: $\quad (New\mathbf{p}, New\Delta F^+, New\Delta F^-) \leftarrow$ UPDATE($\mathbf{p}, \Delta F^+, \Delta F^-, \delta$)
13: $\quad \Delta F^+ \leftarrow New\Delta F^+$
14: $\quad \Delta F^- \leftarrow New\Delta F^-$
15: $\quad$ **if** $F(New\mathbf{p}) > CurrentF$ **then**
16: $\quad\quad \mathbf{p} \leftarrow New\mathbf{p}$
17: $\quad\quad CurrentF \leftarrow F(New\mathbf{p})$
18: $\quad\quad \alpha \leftarrow random(\alpha, \gamma\alpha)$
19: $\quad$ **else**
20: $\quad\quad \alpha \leftarrow random(\beta_1\alpha, \beta_2\alpha)$
21: $\quad\quad$ **if** $\alpha < \alpha_{tol}$ **then**
22: $\quad\quad\quad flag \leftarrow True$
23: $\quad\quad$ **end if**
24: $\quad$ **end if**
25: $\quad z{+}{+}$
26: **end while**

Algorithm 3 Initialization Function

1: **function** INITIALIZATION(δ)
2: $\quad p_{ik} \leftarrow P_k^L + \delta$ for any $i = 1, \dots, N, k \in J_i$
3: $\quad$ Compute $t^*(\mathbf{p})$ by Theorem 4.1
4: $\quad$ **while** constraints (8), (9), (10) are not satisfied **do**
5: $\quad\quad$ **for** $k \in J$ **do**
6: $\quad\quad\quad$ **if** $\sum_{i:\ k \in J_i} t_{ik}^*(\mathbf{p}) < t_k^L$ **then**
7: $\quad\quad\quad\quad$ **repeat**
8: $\quad\quad\quad\quad\quad p_{ik} \leftarrow p_{ik} + \delta$ for any i such that $k \in J_i$
9: $\quad\quad\quad\quad\quad$ Compute $t^*(\mathbf{p})$
10: $\quad\quad\quad\quad$ **until** $\sum_{i:\ k \in J_i} t_{ik}^*(\mathbf{p}) \ge t_k^L$
11: $\quad\quad\quad$ **end if**
12: $\quad\quad$ **end for**
13: $\quad\quad$ **if** $\sum_{i=1}^{N} \sum_{k \in J_i} p_{ik} t_{ik}^*(\mathbf{p}) > B$ **then**
14: $\quad\quad\quad$ **repeat**
15: $\quad\quad\quad\quad p_{ik} \leftarrow p_{ik} - \delta$ for any $i = 1, \dots, N, k \in J_i$
16: $\quad\quad\quad\quad$ Compute $t^*(\mathbf{p})$
17: $\quad\quad\quad$ **until** $\sum_{i=1}^{N} \sum_{k \in J_i} p_{ik} t_{ik}^*(\mathbf{p}) \le B$
18: $\quad\quad$ **end if**
19: $\quad$ **end while**
20: $\quad$ **for** $i = 1, \dots, N$ **do**
21: $\quad\quad$ **for** $k \in J_i$ **do**
22: $\quad\quad\quad \mathbf{p}' \leftarrow \mathbf{p}$
23: $\quad\quad\quad p_{ik}' \leftarrow p_{ik} + \delta$
24: $\quad\quad\quad \Delta F_{ik}^+ \leftarrow \dfrac{F(\mathbf{p}') - F(\mathbf{p})}{\delta}$
25: $\quad\quad\quad \mathbf{p}' \leftarrow \mathbf{p}$
26: $\quad\quad\quad p_{ik}' \leftarrow p_{ik} - \delta$
27: $\quad\quad\quad \Delta F_{ik}^- \leftarrow \dfrac{F(\mathbf{p}') - F(\mathbf{p})}{\delta}$
28: $\quad\quad$ **end for**
29: $\quad$ **end for**
30: $\quad$ Sort lists ΔF^+ and ΔF^- decreasingly
31: **return** $(\mathbf{p}, \Delta F^+, \Delta F^-)$
32: **end function**

F at the found price vector $\mathbf{p}$ along any direction. The lists ΔF^+ and ΔF^- are then sorted in descending order.

At each iteration, Algorithm 2 sets the step size δ (line 11) and find a new feasible price vector by means of the Update function (see Algorithm 4). The Update function is inspired to the directional direct-search algorithm that is a well known solution approach in the field of derivative-free optimization [58]. The Update function exploits the lists ΔF^+ and ΔF^- to get the new price vector. The two lists contain, at each iteration, approximate values of the incremental ratios of F at the current price with the current step size along any direction. The Update function starts by changing only the price that (hopefully) provides the maximum platform net utility increase. If this maximum is achieved at ΔF^+_{ik}, for some user u_i and job k, then the price p_{ik} is increased by δ (line 11), otherwise it is decreased by δ (line 23). Then, if the new price vector $\mathbf{p'}$ is feasible and provides a platform net utility greater than the current one (line 12 or 24), then the procedure stops, returns $\mathbf{p'}$ (line 13 or 25) and update the value of ΔF^-_{ik} (if p_{ik} has been increased by δ, see line 14) or ΔF^+_{ik} (if p_{ik} has been decreased by δ, see line 26). Otherwise, if the new price $\mathbf{p'}$ is not feasible or it does not provide a platform net utility greater than the current one, then the values of ΔF^+_{ik} or ΔF^-_{ik} are updated (line 17 or 29) and the next element in the list ΔF^+ or ΔF^- will be analysed (lines 18 or 30).

If the price vector $New\mathbf{p}$ returned by the Update function gives a net utility greater than the current one, then the iteration is considered successful and the step size δ is increased (Algorithm 2 lines 16-18), otherwise it is considered unsuccessful and δ is decreased (line 20). Algorithm 2 stops when either the step size is smaller than a given threshold (lines 21-23) or the maximum number of iterations is reached.

As discussed in [58], a too large step size can cause the algorithm to overstep a local optimum, while a large enough step size leads to a fast convergence. On the other hand, a small step size will take a long time to converge. Therefore, we use a large step size at first iterations, then if the platform net utility does not improve, it means that the step size should be decreased. Vice versa, if the platform net utility increases, a δ increase in next iteration can lead the algorithm to converge faster. By this argument, we can keep the advantage of both large and small step sizes while the search advances. The impact of the variable step size adoption on the convergence of Algorithm 2 and a comparison with a fixed step size approach will be discussed in Section 7.2.3.

Algorithm 4 Update Function

1: **function** UPDATE($\mathbf{p}, \Delta F^+, \Delta F^-, \delta$)
2: $New\mathbf{p} \leftarrow \mathbf{p}$
3: $New\Delta F^+ \leftarrow \Delta F^+$
4: $New\Delta F^- \leftarrow \Delta F^-$
5: $flag \leftarrow False,\ r \leftarrow 1,\ s \leftarrow 1$
6: **while** ($flag = False$) AND ($r \leq |\Delta F^+|$ OR $s \leq |\Delta F^-|$) **do**
7: $M \leftarrow$ max between r-th elem. of ΔF^+ and s-th elem. of ΔF^-
8: **if** M is equal to the r-th element of ΔF^+ **then**
9: Let ΔF^+_{ik} be the r-th element of ΔF^+
10: $\mathbf{p'} \leftarrow \mathbf{p}$
11: $p'_{ik} \leftarrow p_{ik} + \delta$
12: **if** $\mathbf{p'}$ satisfies (8), (9), (10) AND $F(\mathbf{p'}) > F(\mathbf{p})$ **then**
13: $New\mathbf{p} \leftarrow \mathbf{p'}$
14: $New\Delta F^-_{ik} \leftarrow \dfrac{F(\mathbf{p}) - F(\mathbf{p'})}{\delta}$
15: $flag \leftarrow True$
16: **else**
17: $New\Delta F^+_{ik} \leftarrow \dfrac{F(\mathbf{p'}) - F(\mathbf{p})}{\delta}$
18: $r{+}{+}$
19: **end if**
20: **else**
21: Let ΔF^-_{ik} the s-th element of ΔF^-
22: $\mathbf{p'} \leftarrow \mathbf{p}$
23: $p'_{ik} \leftarrow p_{ik} - \delta$
24: **if** $\mathbf{p'}$ satisfies (8), (9), (10) AND $F(\mathbf{p'}) > F(\mathbf{p})$ **then**
25: $New\mathbf{p} \leftarrow \mathbf{p'}$
26: $New\Delta F^+_{ik} \leftarrow \dfrac{F(\mathbf{p}) - F(\mathbf{p'})}{\delta}$
27: $flag \leftarrow True$
28: **else**
29: $New\Delta F^-_{ik} \leftarrow \dfrac{F(\mathbf{p'}) - F(\mathbf{p})}{\delta}$
30: $s{+}{+}$
31: **end if**
32: **end if**
33: **end while**
34: Sort lists $New\Delta F^+$ and $New\Delta F^-$ decreasingly
35: **return** ($New\mathbf{p}$,$New\Delta F^+$,$New\Delta F^-$)
36: **end function**

7. Performance evaluation

In this section, we present numerical experiments to evaluate the performance of our approach. The experimental setup including the characteristics of the sensing platform and mobile users is introduced in Section 7.1. The scalability analysis and the quality of the solutions that can be achieved by our approach are reported in Section 7.2.
All experiments were run on a Linux server machine with 8-cores Intel(R) Xeon(R) CPU 2.40GHz and 16 GB memory. We used multiprocessing to support the solution of both the Stackelber game reformulation proposed in Section 5 (enabling the multithreading option for the BARON 19.12.7 solver) and the heuristic approach presented in Section 6.

7.1. Experiments setup

In our experiments, we consider a sensing area where mobile users are randomly distributed. Even if the solutions we propose are general and can cope with an arbitrary number of jobs, to perform a fair comparison with the work proposed in [1], in a first scenario we assume that there are two types of jobs $J = \{1, 2\}$. Furthermore, we consider a second scenario including three types of jobs $J = \{1, 2, 3\}$. Due to the unavailability of the parameters used even in very popular MCS applications such as Bemyeyes [22] or Pokemon Go [23], the parameters of the sensing platform utility functions and mobile users' cost functions are generated randomly as in previous literature proposals [1, 36, 37, 38, 50, 51, 52, 53] and distributed as reported in Table 2. To have a fair comparison with [1], we choose the same range of the random parameters in this previous work. To assess the performance in a wide set of scenarios of practical interest, the comparison is performed

Table 2
Simulation parameters

Parameter	Value
N	10, 100, 500, 1000, 5000
ω_{ik}	Uniform distribution in (0, 1)
a_{ik}	Uniform distribution in (1, 2)
b_{ik}	Uniform distribution in (0.5, 1)
c_{ik}	0
T_i	Uniform distribution in (2, 3)
μ_k	10, 30, 50
B	equal to N
P_k^U	5
P_k^L	0.5
t_k^L	0.3
t_k^U	3
α_{tol}	0.0001

for both small and large systems. The number of users has been set equal to $N = 10, 100, 500, 1000$, and 5000. For a fixed problem instance size, we generated 20 random instances and in the following every method relevant metrics are evaluated as the average across 20 instances.

For what concerns the connectivity of the mobile users, we assume that the users participating to the sensing jobs are connected through 4G or 5G networks. As network performance metric, we consider only the number of exchanged messages. Indeed, both the work in [1] and our approach require to exchange a single float value in the interaction between the provider and each mobile user. In [1], at each iteration the platform solves the primal problem and sends to each mobile user the current value of the Lagrangian multipliers of the objective function of the primal problem which are the decision variables of the corresponding dual problem. Vice versa, each mobile user solves a local dual sub-problem and sends back to the platform the time she/he is willing to devote to the jobs. The time required to send and receive a single float number can be estimated, as a first approximation, by the round trip time of the messages that ranges between 50-100 ms for 4G networks (see, e.g., [59]) and has a target value of 1 ms for the 5G standard [60, 61]. As it will be discussed in Section 7.2.1, network transfer times are significantly larger than the computation time required to solve the local sub-problems (in the order of ms). Hence, the number of exchanged messages is a representative metric to compare our solution with the approach in [1].

Overall in the comparison with the dual decomposition approach, we performed 240 simulations which required about 700 CPU hours in our server.

7.2. Experimental results

In this section, we compare our heuristic algorithm with the dual decomposition approach proposed in [1] and the BARON solver [43] (which has been used to solve the reformulation reported in Section 5) in Sections 7.2.1 and 7.2.2, respectively. The impact of the dynamic step size on the convergence of our algorithm is analyzed in Section 7.2.3.

7.2.1. Comparison with the dual decomposition approach

To validate our approach, we compare the performance of our heuristic method with the dual decomposition approach proposed in [1]. KNITRO 12.0 [62] has been used as nonlinear optimization solver for the solution of the [1] optimization problem on the platform side and GUROBI 9.0.2 [63] as a quadratic program solver for the solution of the [1] optimization problem on the user side. In the dual decomposition method we have used the multithreading option for KNITRO to solve the platform side optimization problem. The comparison considers as performance metrics: i) the number of exchanged messages, ii) the running time, and iii) the platform net utility. To have a fair comparison, since the approach in [1] do not consider constraint families (8), (9) and (10), they have been relaxed. Here we limit our results to $N = 1000$ users, since instances of that size required more than one million messages to converge with the dual decomposition approach. The results are as follows:

- **Number of messages**
  The average number of messages sent among the users and the platform (across 20 random instances for each user set considering two types of jobs) are shown in Figure 3. As can be seen, the number of messages in the dual decomposition approach is significantly larger than in our method. Indeed, the dual decomposition approach (since the platform is not aware of the users' parameters) has to solve a distributed optimization problem and this leads to a large number of messages to update the Lagrangian multipliers and users' job times. Vice versa, with our parameter estimation algorithm, the platform estimates users' parameters within the initialization phase after sending and receiving a few messages. Then, after estimating the parameters by solving the 2x2 linear equation system (16), the platform can solve the problem in a centralized way by the proposed heuristic algorithm. The experiments show that our method allows to reduce the average number of messages by a factor ranging between 53 (for 10 users) and 80 (for 1000 users). Hence, our solution approach is more scalable than the work proposed in [1], since the wide area network is the bottleneck in the reference distributed systems (indeed, the round trip time between a user device and the platform is higher than the time required to compute Lagrangian multiplier updates in [1], see also the next point). Note that the number of network messages reduction increases with the number of users.

- **Running time**
  According to the dual decomposition distributed algorithm proposed in [1], at each iteration the platform sends a message including the current Lagrangian multiplier value to all users. On the other side, the users solve their dual local optimization problem and send their optimal time as a result to the platform. Users can

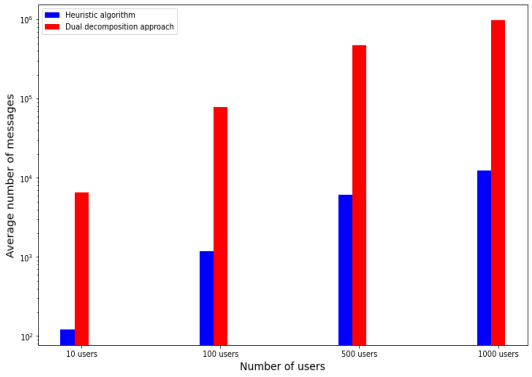


Figure 3: Average number of messages comparison between the dual decomposition approach and our heuristic algorithm.

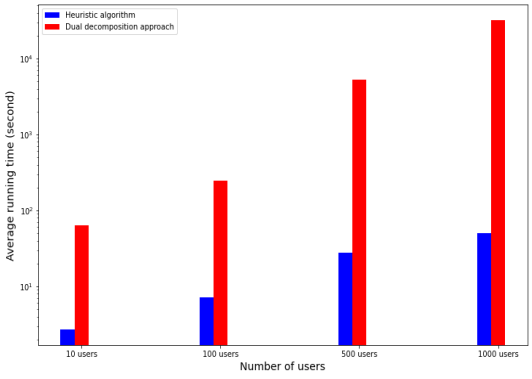


Figure 4: Average running time comparison between the dual decomposition approach and our heuristic algorithm.

solve their optimization problem in a parallel manner. Therefore, since the platform updates are performed synchronously, [1] overall performance will be affected by the time of the last message received by the platform at each iteration.

Since wide area networks round trip times are usually characterised by large variability at least in current 4G networks [59] and we have already shown that our approach reduces the number of exchanged messages, for comparison purposes at each iteration performed by [1], we neglect the network round trip time and we just consider the computation time of the platform and the maximum computation time of users.

The average running time for the different number of users N are shown in Figure 4. The experiments show that the adoption of our heuristic method can reduce the average running time significantly, by more than 23 times for 10 users up to more than 650 times for 1000 users. Therefore, even neglecting the network delay and just considering the computation time, our approach outperforms the dual decomposition method in [1].

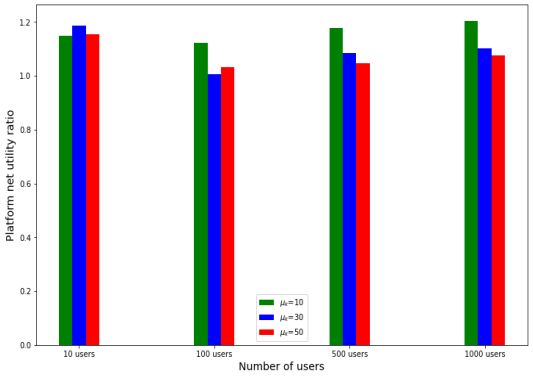


Figure 5: Platform net utility ratio of our heuristic algorithm with respect to the dual decomposition approach.

- **Platform Net Utility**
  Figure 5 shows a comparison between the ratio of platform net utility in the heuristic method to the one in dual decomposition approach for different value of μ equal to 10, 30 and 50. Since the platform is the game leader, our approach achieves a platform net utility improvement between 0.6% and 20.5% on average.

As final remark, the experimental results show that our Stackelberg game approach and the heuristic method excel with respect to [1] according to every performance metric of interest: not only we reduce the computation time by at least a factor 23 and the number of messages by at least a factor 53, but we also obtain a platform net utility increase.

As a final remark, in our work (as in most of distributed approaches such as [1, 36]), we assume that the end users will not move from the sensing job area during all steps of the MCS. According to Figure 4, in our centralized approach, for 1000 user (i.e., a scenario of significant size), the users have to stay about 52 seconds in the job coverage area to receive the optimal time and price that is reasonable in practice, while in the dual decomposition approach proposed in [1], they have to stay about 9 hours.

7.2.2. Comparison with the BARON solver

In this section, we compare the performance of the heuristic algorithm with the solution of the problem reformulation discussed in Section 5, which can be achieved by the BARON solver in both scenarios of two and three jobs. BARON is a state of the art commercial global solver for Mixed-Integer Nonlinear Programs [43]. Since the problem (17)–(28) has a non-concave objective function and includes non-convex constraints, only relying on local search procedures, as the direct-search, might lead to poor local sub-optima. BARON, as a global solver, overcomes this weakness through interval analysis and range reduction techniques (i.e., linear programming, duality, and constraint propagation) within a branch-and-bound framework to find global solutions to non-convex models.

In this experiment, we reintroduce the price, budget and time constraints for the platform i.e., constraints (8), (9) and (10). To have a fair comparison, we have also used multi-threading to run both our heuristic algorithm and the BARON solver.

- **Two jobs scenario**
  The average results achieved by considering 10 random instances and by varying the number of users N for the two jobs scenario are shown in Figure 6. The figure reports the average platform net utility versus the execution time. It can be noticed that the average of platform net utility in our proposed algorithm is near to BARON's result. For 10 users, the heuristic algorithm's result is about 2.2% larger than BARON but when the number of users N is in the range [100, 5000], BARON achieves a solution with a platform net utility by 0.6% and 2.9% better than the heuristic algorithm. According to Figure 6, our heuristic algorithm can reach an acceptable result in few seconds. The dotted line in the figures for BARON's result means that BARON can not find a feasible solution in that time limit. Hence, our algorithm acts much faster than BARON especially for large scale systems: for 1000 users our algorithm can find a good solution after about 5 seconds, while BARON can find a feasible solution near to optimal after about 39 seconds. For 5000 users, the heuristic algorithm can find a good solution after 8 seconds, whereas BARON can find feasible solutions for 9 instances after 258 seconds, while for one out of 10 instances the solution cannot be found in 1000 seconds.

- **Three jobs scenario**
  The average results achieved by considering 10 random instances and by varying the number of users N for the three jobs scenario are shown in Figure 7. Results show that the average of the platform net utility obtained by BARON and our proposed algorithm are very close. Up to 1000 users the heuristic algorithm's result is in the very worst case lower than 2% w.r.t. BARON while for 5000 users results are about 1.4% larger on average. As it can be seen, our algorithm acts much faster than BARON especially for large scale systems. According to Figure 7b, for 100 users BARON can find a feasible solution only for two out of 10 instances after about one second, while our algorithm can find a near-optimal solution for all instances after about 0.5 second. As it can be seen in Figure 7c, for 500 users BARON can find a feasible solution for five out of 10 instances after about 19 seconds, whereas our algorithm find a near-optimal solution for all instances after about two seconds.

  Finally, for 5000 users, our algorithm can find the optimal solution after 13 seconds, whereas BARON can find a lower solution after 570 seconds.

7.2.3. Impact of dynamic step size on the convergence

In this section, we compare the impact of the dynamic step size and fixed step size on the convergence of our algorithm. In particular, we set $\alpha_{tol} = 0.0001$ for the variable version, while we set $\delta_{fixed} = 0.1, 0.2, 0.3, 1$ and 2 in the fixed step version. In this way, we investigated multiple values of δ_{fixed} which corresponds to the values assumed by δ in the variable step size runs. As an example, a representative trace for 100 users is depicted in Figure 8. It can be noticed that the dynamic step size not only can achieve a better result but also can be faster than the fixed step size version.

8. Conclusions

In this paper, we have designed a framework and an incentive mechanism based on a Stackelberg game for mobile crowdsensing systems. Different from other literature proposals, we have introduced a budget constraint for the platform and developed a centralized method for the game solution since, thanks to KKT conditions, we can estimate the users utility function parameters. Therefore, the platform is able to solve its optimization problem in a centralized manner instead of exchanging a lot of messages as requested by distributed methods. Furthermore, we have proposed an efficient heuristic algorithm to achieve good approximated solutions of the game. Results have demonstrated that our approach outperforms current literature proposals and is much faster than the BARON solver especially in large scale systems.

Future works will extend our method to general utility functions for the end users (not necessarily quadratic) and develop models where users can switch between 4G/5G and Wi-Fi connections. Finally, scenarios where a fraction of users participating to the game are not truthful will be also investigated, proposing some protection mechanisms for the platform provider.

APPENDIX

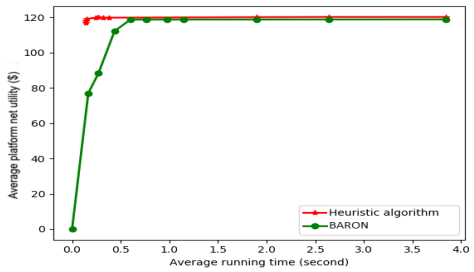
A. Proof of Theorem 4.1

Proof. Problem (11)–(13) has a strongly concave objective function and linear constraints, thus its unique solution is the unique solution of the corresponding Karush-Kuhn-Tucker (KKT) system:

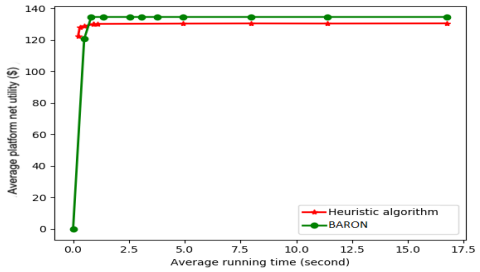
$$
\begin{aligned}
& a_{ik} t_{ik} - d_{ik} + \lambda - \mu_k = 0, \qquad k \in J_i, \\
& \lambda \geq 0, \quad \sum_{k \in J_i} t_{ik} \leq T_i, \quad \lambda \left(\sum_{k \in J_i} t_{ik} - T_i \right) = 0, \\
& t_{ik} \geq 0, \quad \mu_k \geq 0, \quad t_{ik} \mu_k = 0, \quad k \in J_i,
\end{aligned}
$$

for some multipliers λ and μ. The KKT system can be equivalently rewritten as:

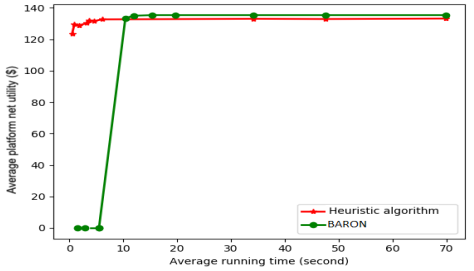
$$
\begin{aligned}
& a_{ik} t_{ik} - d_{ik} + \lambda \geq 0 && k \in J_i, \\
& t_{ik} \geq 0 && k \in J_i,
\end{aligned}
$$



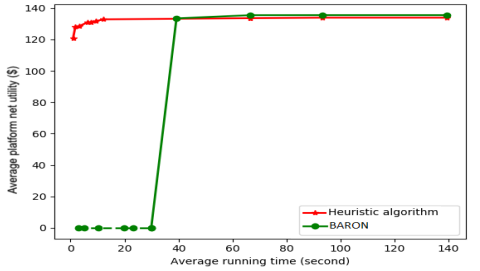
(a) Comparison for 10 users.



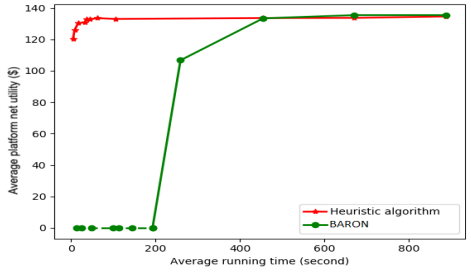
(b) Comparison for 100 users.



(c) Comparison for 500 users.



(d) Comparison for 1000 users.



(e) Comparison for 5000 users.

Figure 6: Comparison between the solutions obtained by BARON solver and the heuristic algorithm varying the number of users in the two jobs scenario.

$$
\begin{aligned}
& t_{ik}(a_{ik}t_{ik} - d_{ik} + \lambda) = 0 \qquad\qquad k \in J_i, \\
& \lambda \geq 0, \\
& \sum_{k \in J_i} t_{ik} \leq T_i, \\
& \lambda \left(\sum_{k \in J_i} t_{ik} - T_i \right) = 0.
\end{aligned}
$$

We identify the following cases:

a) If $k \in J_i^-$, then $t_{ik}^* = 0$ must hold. In fact, if $t_{ik}^* > 0$, then the complementarity slackness condition implies $a_{ik}t_{ik}^* - d_{ik} + \lambda = 0$. On the other hand, $d_{ik} \leq 0$ implies that $a_{ik}t_{ik}^* - d_{ik} + \lambda \geq a_{ik}t_{ik}^* > 0$ which is a contradiction.

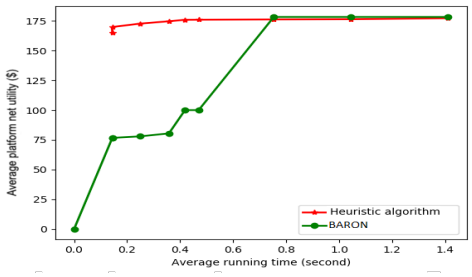
b) Let us consider indices $k \in J_i^+$. We notice that if $\lambda \geq d_n > 0$, then $t_{ik}^* = 0$ for any $k \in J_i^+$ and $\sum_{k \in J_i} t_{ik}^* = T_i$, which is impossible. Therefore, $\lambda < d_n$ holds. We distinguish the following $n+1$ cases: $\lambda = 0$, $\lambda \in (0, d_{i1})$, $\lambda \in [d_{i1}, d_{i2})$, $\ldots$, $\lambda \in [d_{i,n-1}, d_{in})$.

- If $\lambda = 0$, then $a_{ik}t_{ik}^* \geq d_{ik} > 0$ holds for any $k \in J_i^+$, thus $t_{ik}^* > 0$ and, in particular, we get $t_{ik}^* = d_{ik}/a_{ik}$ for any $k = 1, \ldots, n$. If $\sum_{k=1}^{n} \frac{d_{ik}}{a_{ik}} \leq T_i$, then $(t_{ik}^*)_{k \in J_i^+}$ solves the KKT system, with $\lambda = 0$, and hence solves problem (11)–(13) as well.
- If $\lambda \in (0, d_{i1})$, then $a_{ik}t_{ik}^* \geq d_{ik} - \lambda > 0$ holds for any $k \in J_i^+$, thus $t_{ik}^* > 0$ and $t_{ik}^* = (d_{ik} - \lambda)/a_{ik}$. Since $\lambda > 0$, we have

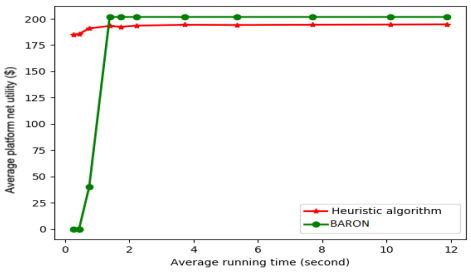
$$
T_i = \sum_{k \in J_i} t_{ik}^* = \sum_{k=1}^{n} \frac{d_{ik}}{a_{ik}} - \lambda \sum_{k=1}^{n} \frac{1}{a_{ik}},
$$

thus we have

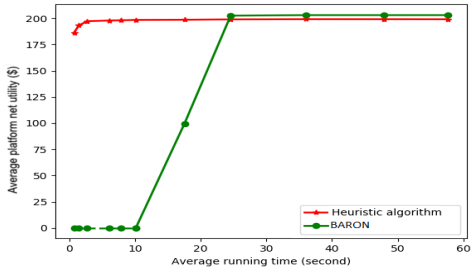
$$
\lambda = \frac{\sum_{k=1}^{n} \frac{d_{ik}}{a_{ik}} - T_i}{\sum_{k=1}^{n} \frac{1}{a_{ik}}} = z_1.
$$



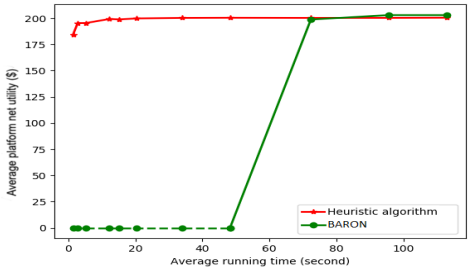
(a) Comparison for 10 users.



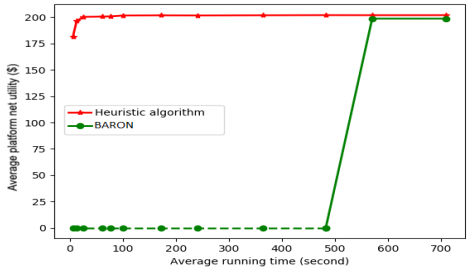
(b) Comparison for 100 users.



(c) Comparison for 500 users.



(d) Comparison for 1000 users.



(e) Comparison for 5000 users.

Figure 7: Comparison between the solutions obtained by BARON solver and the heuristic algorithm varying the number of users in the three jobs scenario.

Therefore, if $z_1 \in (0, d_{i1})$, then $t^*_{ik} = \dfrac{d_{ik} - z_1}{a_{ik}}$, for any $k = 1, \dots, n$, is a solution of the KKT system, with $\lambda = z_1$, and thus solves problem (11)–(13). □

- If $q \in \{2, \dots, n\}$ and $\lambda \in [d_{i,q-1}, d_{iq})$, then

$$t^*_{ik} = \begin{cases} 0 & \text{if } k = 1, \dots, q-1, \\ \dfrac{d_{ik} - \lambda}{a_{ik}} & \text{if } k = q, \dots, n. \end{cases} \tag{29}$$

Since $T_i = \sum_{k=1}^{n} t^*_{ik}$, we get

$$\lambda = \frac{\displaystyle\sum_{k=q}^{n} \frac{d_{ik}}{a_{ik}} - T_i}{\displaystyle\sum_{k=q}^{n} \frac{1}{a_{ik}}} = z_q.$$

Therefore, if $z_q \in [d_{i,q-1}, d_{iq})$, then the vector defined in (29) is a solution of the KKT system, with $\lambda = z_q$, and thus solves problem (11)–(13).

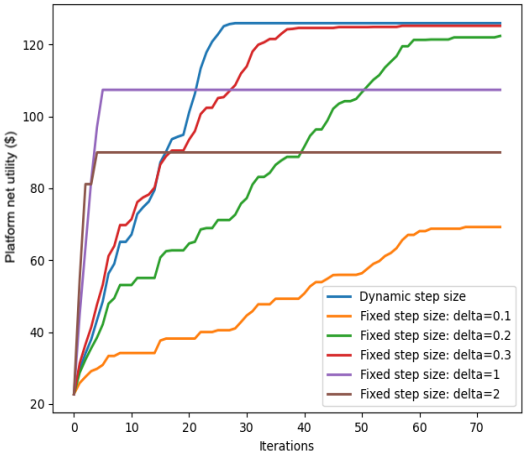


Figure 8: Platform net utility of dynamic step size vs fixed step size for 100 users.

B. Optimal solution of the user problem with three jobs

We report here the optimal solution in closed form of problem (11)–(13) with three jobs solved by user i. For the sake of simplicity, we remove index i in the following formulas. The optimal solution is as follows:

$$
t^* = \begin{cases}
\left(\frac{p_1-b_1}{a_1}, \frac{p_2-b_2}{a_2}, \frac{p_3-b_3}{a_3}\right) & \text{if } p_1 \geq b_1, p_2 \geq b_2, p_3 \geq b_3, \sum_{k=1}^{3} \frac{p_k-b_k}{a_k} \leq T, \\
\left(0, \frac{p_2-b_2}{a_2}, \frac{p_3-b_3}{a_3}\right) & \text{if } p_1 \leq b_1, p_2 \geq b_2, p_3 \geq b_3, \frac{p_2-b_2}{a_2} + \frac{p_3-b_3}{a_3} \leq T, \\
\left(\frac{p_1-b_1}{a_1}, 0, \frac{p_3-b_3}{a_3}\right) & \text{if } p_1 \geq b_1, p_2 \leq b_2, p_3 \geq b_3, \frac{p_1-b_1}{a_1} + \frac{p_3-b_3}{a_3} \leq T, \\
\left(\frac{p_1-b_1}{a_1}, \frac{p_2-b_2}{a_2}, 0\right) & \text{if } p_1 \geq b_1, p_2 \geq b_2, p_3 \leq b_3, \frac{p_1-b_1}{a_1} + \frac{p_2-b_2}{a_2} \leq T, \\
\left(\frac{a_3(p_1-b_1-p_2+b_2)+a_2(p_1-b_1-p_3+b_3)+a_2a_3T}{a_1a_2+a_1a_3+a_2a_3}, \right. & \text{if } T \leq \sum_{k=1}^{3} \frac{p_k-b_k}{a_k} \leq T + \left(\sum_{k=1}^{3} \frac{1}{a_k}\right) \min_{k=1,2,3}\{p_k-b_k\}, \\
\quad \frac{a_3(p_2-b_2-p_1+b_1)+a_1(p_2-b_2-p_3+b_3)+a_1a_3T}{a_1a_2+a_1a_3+a_2a_3}, & \\
\quad \left. \frac{a_2(p_3-b_3-p_1+b_1)+a_1(p_3-b_3-p_2+b_2)+a_1a_2T}{a_1a_2+a_1a_3+a_2a_3}\right) & \\
(0,0,0) & \text{if } p_1 \leq b_1, p_2 \leq b_2, p_3 \leq b_3, \\
(0,0,T) & \text{if } p_3 - b_3 - a_3T \geq \max\{0, p_1-b_1, p_2-b_2\}, \\
(0,T,0) & \text{if } p_2 - b_2 - a_2T \geq \max\{0, p_1-b_1, p_3-b_3\}, \\
(T,0,0) & \text{if } p_1 - b_1 - a_1T \geq \max\{0, p_2-b_2, p_3-b_3\}, \\
\left(0,0,\frac{p_3-b_3}{a_3}\right) & \text{if } p_1 \leq b_1, p_2 \leq b_2, p_3 \in [b_3, b_3+a_3T], \\
\left(0,\frac{p_2-b_2}{a_2},0\right) & \text{if } p_1 \leq b_1, p_3 \leq b_3, p_2 \in [b_2, b_2+a_2T], \\
\left(\frac{p_1-b_1}{a_1},0,0\right) & \text{if } p_2 \leq b_2, p_3 \leq b_3, p_1 \in [b_1, b_1+a_1T], \\
\left(0, \frac{p_2-b_2-p_3+b_3+a_3T}{a_2+a_3}, \frac{p_3-b_3-p_2+b_2+a_2T}{a_2+a_3}\right) & \text{if } \frac{p_2-b_2}{a_2} + \frac{p_3-b_3}{a_3} \geq \max\left\{T, T+(p_1-b_1)\left(\frac{1}{a_2}+\frac{1}{a_3}\right)\right\}, \\
& \quad p_2-b_2 \in [p_3-b_3-a_3T, p_3-b_3+a_2T], \\
\left(\frac{p_1-b_1-p_3+b_3+a_3T}{a_1+a_3}, 0, \frac{p_3-b_3-p_1+b_1+a_1T}{a_1+a_3}\right) & \text{if } \frac{p_1-b_1}{a_1} + \frac{p_3-b_3}{a_3} \geq \max\left\{T, T+(p_2-b_2)\left(\frac{1}{a_1}+\frac{1}{a_3}\right)\right\}, \\
& \quad p_3-b_3 \in [p_1-b_1-a_1T, p_1-b_1+a_3T], \\
\left(\frac{p_1-b_1-p_2+b_2+a_2T}{a_1+a_2}, \frac{p_2-b_2-p_1+b_1+a_1T}{a_1+a_2}, 0\right) & \text{if } \frac{p_1-b_1}{a_1} + \frac{p_2-b_2}{a_2} \geq \max\left\{T, T+(p_3-b_3)\left(\frac{1}{a_1}+\frac{1}{a_2}\right)\right\}, \\
& \quad p_1-b_1 \in [p_2-b_2-a_2T, p_2-b_2+a_1T].
\end{cases}
$$

References

[1] Yufeng Zhan, Yuanqing Xia, and Jinhui Zhang. Quality-aware incentive mechanism based on payoff maximization for mobile crowdsensing. *Ad Hoc Networks*, 72:44–55, 2018.

[2] Jiaoyan Chen and Jingsen Yang. Maximizing coverage quality with budget constrained in mobile crowd-sensing network for environmental monitoring applications. *Sensors*, 19(10):1–17, 2019.

[3] Rosa Ma Alsina-Pagès, Unai Hernandez-Jayo, Francesc Alías, and Ignacio Angulo. Design of a mobile low-cost sensor network using urban buses for real-time ubiquitous noise monitoring. *Sensors*, 17(1), 2017.

[4] Marco Zappatore, Antonella Longo, and Mario A. Bochicchio. Using mobile crowd sensing for noise monitoring in smart cities. In *International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*, 2016.

[5] Marco Zappatore, Antonella Longo, and Mario A. Bochicchio. A crowdsensing approach for mobile learning in acoustics and noise monitoring. In *In Proceedings of the ACM Symposium on Applied Computing*, 2016.

[6] Moustafa Elhamshary, Moustafa Youssef, Akira Uchiyama, Hirozumi Yamaguchi, and Teruo Higashino. Transitlabel: A crowd-sensing system for automatic labeling of transit stations semantics. In *In Proceedings of the International Conference on Mobile Systems*, 2016.

[7] Heba Aly, Anas Basalamah, and Moustafa Youssef. Map++: A crowd-sensing system for automatic map semantics identification. In *In Proceedings of the Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2014.

[8] Yao-Chung Fan, Wei Hong Lee, Cheng Teng Iam, and Gia Hao Syu. Indoor place name annotations with mobile crowd. In *International Conference on Parallel and Distributed Systems*. IEEE, 2013.

[9] Selek Ceren Celik and Ozlem Durmaz Incel. Semantic place prediction from crowd-sensed mobile phone data. *Journal of Ambient Intelligence and Humanized Computing*, 9:2109–2124, 2018.

[10] Heba Aly, Anas Basalamah, and Moustafa Youssef. Automatic rich map semantics identification through smartphone-based crowd-sensing. *IEEE Transactions on Mobile Computing*, 16(10):2712–2725, 2017.

[11] Xiao Li and Daniel W.Goldberg. Toward a mobile crowdsensing system for road surface assessment. *Computers, Environment and Urban Systems*, 69:51–62, 2018.

[12] Sultan Basudan, Xiaodong Lin, and Karthik Sankaranarayanan. A privacy-preserving vehicular crowdsensing based road surface condition monitoring system using fog computing. *IEEE INTERNET OF THINGS JOURNAL*, 4(3):772–782, 2017.

[13] Yao-Chung Fan, Wei Hong Lee, Cheng Teng Iam, and Gia Hao Syu. Towards on demand road condition monitoring using mobile phone sensing as a service. In *Procedia Computer science*, volume 83, pages 345–352, 2016.

[14] Lingzhi Yi, Xianjun Deng, Minghua Wang, Dexin Ding, and Yan Wang. Localized confident information coverage hole detection in internet of things for radioactive pollution monitoring. *IEEE Transactions on Mobile Computing*, 5(5):18665–18674, 2017.

[15] Lingzhi Yi, XianjunDeng, ZenghuiZou, DexinDing, and Laurence T.Yang. Confident information coverage hole detection in sensor networks for uranium tailing monitoring. *Journal of Parallel and Distributed Computing*, 118(1):57–66, 2018.

[16] Thiagarajan Arvindvand Ravindranath Lenin, LaCurts Katrina Leigh, Madden Samuel R.and Balakrishnan Hariand Toledo Sivan, and Eriksson Jakob. Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, page 85–98, 2009.

[17] Vili Lehdonvirta, Yefeng Liu, Mieke Kleppe, Todorka Alexandrova, Hiroaki Kimura, and Tatsuo Nakajima. A crowdsourcing based mobile image translation and knowledge sharing service. In *In Proceedings of the International Conference on Mobile & Ubiquitous Multimedia*, 2010.

[18] Yibo Wu, Yi Wang, Wenjie Hu, and Guohong Cao. Smartphoto: A resource-aware crowdsourcing approach for image sensing with smartphones. *IEEE Transactions on Mobile Computing*, 15(5):1249–1263, 2016.

[19] Murat Demirbas, Murat Ali Bayir, Cuneyt Gurcan Akcora, Yavuz Selim Yilmaz, and Hakan Ferhatosmanoglu. Crowd-sourced sensing and collaboration using twitter. In *IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2010.

[20] Wazir Zada Khan, Yang Xiang, Mohammed Y Aalsalem, and Quratulain Arshad. Mobile phone sensing systems: A survey. *IEEE Communications Surveys & Tutorials*, 15(1):402 – 427, 2012.

[21] Hui Gao, Chi Harold Liu, Wendong Wang, Jianxin Zhao, Zheng Song, Xin Su, Jon Crowcroft, and Kin K. Leung. A survey of incentive mechanisms for participatory sensing. *IEEE Communications Surveys & Tutorials*, 28(7):918–943, 2015.

[22] *Bemyeyes MCS Application*. https://www.bemyeyes.com/.

[23] *Pokemon Go MCS Application*. https://www.Pokemon.com/us/app/pokemon-go/.

[24] The iPhone 12 now has LiDAR, but Google could do it bigger and better'. https://www.androidcentral.com/ipad-pro-has-lidar-camera-should-tech-come-android, October 2020.

[25] Salil S. Kanhere. Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces. In *IEEE 12th International Conference on Mobile Data Management*, pages 415–420. IEEE, 2011.

[26] Jiming Chen, Xianghui Cao, Peng Cheng, Yang Xiao, and Youxian Sun. Distributed collaborative control for industrial automation with wireless sensor and actuator networks. *IEEE Transactions on Industrial Electronics*, 57(12):4219–4230, 2010.

[27] Xianghui Cao, Jiming Chen, Yang Xiao, and Youxian Sun. Building-environment control with wireless sensor and actuator networks: Centralized versus distributed. *IEEE Transactions on Industrial Electronics*, 57(11):3596–3605, 2010.

[28] Shibo He, Dong-Hoon Shin, Junshan Zhang, Jiming Chen, and Youxian Sun. Full-view area coverage in camera sensor networks: Dimension reduction and near-optimal solutions. *IEEE Transactions on Vehicular Technology*, 65(9):7448–7461, 2016.

[29] Luis G. Jaimes, Idalides Vergara-Laurens, and Miguel A. Labrador. A location-based incentive mechanism for participatory sensing systems with budget constraints. In *IEEE International Conference on Pervasive Computing and Communications*, 2012.

[30] Juong-Sik Lee and Baik Hoh. Dynamic pricing incentive for participatory sensing. *Pervasive and Mobile Computing*, 6(6):693–708, 2010.

[31] Juong-Sik Lee and Baik Hoh. Sell your experiences: a market mechanism based incentive for participatory sensing. In *IEEE International Conference on Pervasive Computing and Communications*, 2010.

[32] Xinglin Zhang, Zheng Yang, Zimu Zhou, Haibin Cai, Lei Chen, and Xiangyang Li. Free market of crowdsourcing: Incentive mechanism design for mobile sensing. *IEEE Transactions on Parallel and Distributed Systems*, 25(12):3190–3200, 2014.

[33] Zhibo Wang, Ran Tan, Jiahui Hu, Jing Zhao, Qian Wang, Feng Xia, and Xiaoguang Niu. Heterogeneous incentive mechanism for time-sensitive and location-dependent crowdsensing networks with random arrivals. *Computer Networks*, 131:96–109, 2018.

[34] Jia Peng, Yanmin Zhu, Wei Shu, and Min-You Wu. When data contributors meet multiple crowdsourcers: Bilateral competition in mobile crowdsourcing. *Computer Networks*, 95:1–14, 2016.

[35] Yufeng Zhan, Yuanqing Xia, and Jinhui Zhang. Incentive mechanism in platform-centric mobile crowdsensing: A one-to-many bargaining approach. *Computer Networks*, 132:40–52, 2018.

[36] Xiaoming Duan, Chengcheng Zhao, Shibo He, Peng Cheng, and Junshan Zhang. Distributed algorithms to compute walrasian equilibrium in mobile crowdsensing. *IEEE Transactions on Industrial Electronics*, 64(5):195–209, 2017.

[37] Amir-Hamed Mohsenian-Rad, Vincent W.S. Wong, Juri Jatskevich, and Robert Schober. Optimal and autonomous incentive-based energy consumption scheduling algorithm for smart grid. In *Proc of Innovative Smart Grid Technologies (ISGT)*. IEEE, 2010.

[38] Pedram Samadi, Amir-Hamed Mohsenian-Rad, Robert Schober, Vincent W. S. Wong, and Juri Jatskevich. Optimal real-time pricing algorithm based on utility maximization for smart grid. In *IEEE International Conference on Smart Grid Communications*, pages 415–420. IEEE, 2010.

[39] Zhenni Feng, Yanmin Zhu, Qian Zhang, Lionel M. Ni, and Athanasios V. Vasilakos. Trac: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing. In *Proc of INFOCOM*, pages 1231–1239. IEEE, 2014.

[40] Tong Liu and Yanmin Zhu. Social welfare maximization in participatory smartphonesensing. *Computer Networks*, 73:195–209, 2014.

[41] Shibo He, Dong-Hoon Shin, Junshan Zhang, Jiming Chen, and Phone Lin. An exchange market approach to mobile crowdsensing: Pricing, task allocation and walrasian equilibrium. *IEEE Journal on Selected*

Areas in Communications, 35(4):921–934, 2017.

[42] Wendong Wang, Hui Gao, Chi Harold Liu, and Kin K.Leung. Credible and energy-aware participant selection with limited task budget for mobile crowd sensing. *Ad Hoc Networks*, 43:56–70, 2016.

[43] *MINLP:BARON solver*. https://minlp.com/baron.

[44] Bin Guo, Zhu Wang, Zhiwen Yu, Yu Wang, Neil Y. Yen, Runhe Huang, and Xingshe Zhou. Mobile crowdsensing and computing: the review of an emerging human-powered sensing paradigm. *ACM Computing Surveys*, 48(1):1–31, 2015.

[45] Dejun Yang, Guoliang Xue, Guoliang Xue, Xi Fang, and Jian Tang. Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing. In *Proc. of MobiCom, ACM*, page 173–184, 2012.

[46] Dejun Yang, Guoliang Xue, Xi Fang, and Jian Tang. Incentive mechanisms for crowdsensing: crowdsourcing with smartphones. *IEEE/ACM Transactions on Networking*, 24(3):1732–1744, 2016.

[47] Tie Luo, Salil S. Kanhere, Sajal K. Das, and Hwee-Pink Tan. Incentive mechanism design for heterogeneous crowdsourcing using all-pay contests. *IEEE TRANSACTIONS ON MOBILE COMPUTING*, 15(9):2234–2246, 2016.

[48] Jiangtian Nie, Jun Luo, Zehui Xiong, Dusit Niyato, and Ping Wang. A stackelberg game approach toward socially-aware incentive mechanisms for mobile crowdsensing. *IEEE Transactions on Wireless Communications*, 18(1):724–738, 2019.

[49] Heinrich von Stackelberg. *Market structure and equilibrium*. Springer, Springer, Berlin, Heidelberg, 2011.

[50] Jiangtian Nie, Zehui Xiong, Dusit Niyato, Ping Wang, and Jun Luo. A socially-aware incentive mechanism for mobile crowdsensing service market. In *IEEE Global Communications Conference (GLOBECOM)*, 2018.

[51] Zehui Xiong, Shaohan Feng, Dusit Niyato, Ping Wang, and Yang Zhang. Economic analysis of network effects on sponsored content: A hierarchical game theoretic approach. In *IEEE Global Communications Conference (GLOBECOM)*, 2017.

[52] Ozan Candogan, Kostas Bimpikis, and Asuman Ozdaglar. Optimal pricing in networks with externalities. *Operations Research*, 60(4):883–905, 2012.

[53] Xiaowen Gong, Lingjie Duan, Xu Chen, and Junshan Zhang. When social network effect meets congestion effect in wireless networks: Data usage equilibrium and optimal pricing. *IEEE Journal on Selected Areas in Communications*, 35(2):449–462, 2017.

[54] Shiguang Wang, Dong Wang, Lu Su, Lance Kaplan, and Tarek F. Abdelzaher. Towards cyber-physical systems in social spaces: The data reliability challenge. In *Proc. of RTSS, IEEE*, pages 74–85, 2014.

[55] Lu Su, Qi Li, Shaohan Hu, Shiguang Wang, Jing Gao, Hengchang Liu, Tarek F. Abdelzaher, and Jiawei Han et al. Generalized decision aggregation in distributed sensing systems. In *Proc. of RTSS, IEEE*, pages 1–10, 2014.

[56] Jiming Chen, Weiqiang Xu, Shibo He, Youxian Sun, Preetha Thulasiraman, and Xuemin Shen. Utilitybased asynchronous flow control algorithm for wireless sensor networks. *IEEE J. Sel. Areas Commun*, 17(2):1116–1126, Sep. 2010.

[57] Sean M. Flynn Campbell R. McConnell, Stanley L. Brue. *Microeconomics, PRINCIPLES, PROBLEMS, AND POLICIES*, chapter 9:Businesses and the Costs of Production. McGraw-Hill Education, 2018.

[58] Luis N. Vicente Andrew R. Conn, Katya Scheinberg. *Introduction to Derivative-Free Optimization*. SIAM, 2009.

[59] M. Lauridsen, L. C. Gimenez, I. Rodriguez, T. B. Sorensen, and P. Mogensen. From lte to 5g for connected mobility. *IEEE Communications Magazine*, 55(3):156–162, 2017.

[60] Arvind Narayanan, Eman Ramadan, Jason Carpenter, Qingxu Liu, Yu Liu, Feng Qian, and Zhi-Li Zhang. A first look at commercial 5g performance on smartphones. In *WWW '20: Proceedings of The Web Conference 2020*, page 894–905, 2020.

[61] Marco Ajmone Marsan, Nicola Blefari Melazzi, Stefano Buzzi, and Sergio Palazzo. *The 5G Italy Book 2019: a Multiperspective View of 5G*, chapter 15: Structural Health Monitoring and Earthquake Early Warning over 5G Networks. CNIT:Consorzio Interuniversitario per le Telecomunicazioni, 2019.

[62] *NLP:KNITRO solver*. https://www.artelys.com/solvers/knitro/.

[63] *QP:GUROBI solver*. https://www.gurobi.com/.

Hamta Sedghani received the B.Sc. and M.S degree in Computer Engineering (Software) from Iran University of Since and Technology, Tehran, Iran and university of Tabriz, Tabriz, Iran in 2010 and 2014, respectively. Currently, she is Ph.D. Student in Information Technology Engineering (Computer Networks) and working as a researcher in University of Tabriz. Her current interests include game theory and optimization in mobile crowedsensing.

Danilo Ardagna is Associate Professor at the Dipartimento di Elettronica Informazione and Bioingegneria at Politecnico di Milano. He received a Ph.D. degree in computer engineering in 2004 from Politecnico di Milano from which he also graduated in December 2000. His work focuses on the design, prototype and evaluation of optimization algorithms for resource management of cloud computing and big data systems.

Mauro Passacantando received the M.S. and Ph.D. degrees in Mathematics from University of Pisa, Italy, in 2000 and 2005, respectively. He is currently Associate Professor of Operations Research (qualified for Full Professorship) at the Department of Computer Science at University of Pisa. His research is mainly devoted to variational inequalities and equilibrium problems. In the last years, his work focused on game theoretic models applied to service provisioning problems in cloud and multicloud systems and infrastructure and spectrum sharing in mobile networks.

Mina Zolfy Lighvan received her B.Sc. degree in Computer Engineering (hardware) and M.Sc. degree in Computer Engineering (Computer Architecture) from ECE faculty, university of Tehran, Iran in 1999, 2002 respectively. She received Ph.D. degree in Electronic Engineering (Digital Electronic) from Electrical and Computer Engineering faculty of University of Tabriz, Iran. She is an associate professor in Computer engineering department of ECE faculty in University of Tabriz.

Hadi S. Aghdasi received his B.S. degree in Computer Engineering in 2006 from Sadjad University of Technology, Mashhad, Iran and received his M.S. and Ph.D. degrees in computer engineering from Shahid Beheshti University, Tehran, Iran, in 2008 and 2013, respectively. He is an assistant professor of Computer Engineering department at University of Tabriz, Tabriz, Iran. His current researches focus on Humanoid Robots and Intelligent Methods in Surveillance Systems and Cognitive Technology. Also, he works on Wireless Traditional and Visual Sensor Networks (Routing, Clustering, Coverage, and Visual Information Transmission). He is a director of the both Humanoid Robots and Cognitive Technology (HRCT) and Wireless Ad hoc and Sensor networks (WASL) research laboratories in University of Tabriz.