Optimizing mobile crowdsensing platforms for boundedly rational users

Merkouris Karaliopoulos and Eleni Bakali
Department of Informatics
Athens University of Economics and Business
Athens, Greece
email: {mkaralio, ebakali}@aueb.gr

Abstract—In participatory mobile crowdsensing (MCS) users repeatedly make *choices* among a finite set of alternatives, *i.e.*, whether to contribute to a task or not and which task to contribute to. The platform coordinating the MCS campaigns often *engineers* these choices by selecting MCS tasks to recommend to users and offering monetary or in-kind rewards to motivate their contributions to them. In this paper, we revisit the well-investigated question of how to optimize the contributions of mobile end users to MCS tasks. However, we depart from the bulk of related literature by explicitly accounting for the *bounded rationality* evidenced in human decision making. Bounded rationality is a consequence of cognitive and other kinds of constraints, *e.g.*, time pressure, and has been studied extensively in behavioral science.

We first draw on work in the field of cognitive psychology to model the way boundedly rational users respond to MCS task offers as *Fast-and-Frugal-Trees (FFTs)*. With each MCS task modeled as a vector of feature values, the decision process in FFTs proceeds through sequentially parsing lexicographically ordered features, resulting in choices that are satisfying but not necessarily optimal. We then formulate, analyze and solve the novel optimization problems that emerge for both nonprofit and for-profit MCS platforms in this context. The evaluation of our optimization approach highlights significant gains in both platform revenue and quality of task contributions when compared to heuristic rules that do not account for the lexicographic structure in human decision making. We show how this modeling framework readily extends to platforms that present multiple task offers to the users. Finally, we discuss how these models can be trained, iterate on their assumptions, and point to their implications for applications beyond MCS, where end-users make choices through the mediation of mobile/online platforms.

.

Index Terms—Mobile crowdsensing; incentive allocation; bounded rationality; task recommendation; decision trees; user choice engineering.

1 Introduction

Mobile crowdsensing (MCS) has generated many expectations over the last decade that it will transform the way information is generated and shared among parties interested in it [1], [2]. Technically, it couples the enhanced sensing capabilities of smart mobile devices with a variety of mobile and social technologies, which have made the uploading, processing, and sharing of data easier than ever before. As a result, MCS has been applied in diverse application areas such as environmental monitoring[1], transportation[2], and participatory governance[3].

The implementation of MCS campaigns is more often than not coordinated by a mobile platform and end users typically interact with it through a mobile frontend (app). In this work, we study platforms that *actively* identify and recruit those users who are most suitable for the highly heterogeneous MCS tasks at hand. Consider, for instance, the creation of pollution maps out of smartphone sensors' measurements; or the generation of promotional marketing material for shops out of photos contributed by smartphone users; or the status-tracking of garbage collection points with the help of citizens' reports issued through mobile apps. Users carrying out a task bear some cost in terms of time, cognitive resources, device battery, or physical distance that needs to be covered to perform the task. To outweigh these costs and elicit useful contributions to MCS campaigns, the platform may provide incentives. These incentives are either monetary or in-kind rewards and their efficient allocation to potential contributors demands a good understanding of their particular interests and preferences and how these interact to shape their decisions to contribute (or not) to tasks. Such information, which is the basic ingredient for building *user profiles*, can be collected either implicitly, from historical data about past interactions of users with the MCS platform, or their activity in other social media sites [3]; or, explicitly, through customized questionnaires built in the app.

User profiling goes hand-in-hand with a modeling hypothesis about how users reach decisions, *e.g.*, whether to contribute or not to a given task, and how they make choices, *e.g.*, which task to contribute to out of a set of recommended alternatives. In this respect, the de facto assumption in the current MCS literature is that MCS users behave as *fully rational* agents [4]. Namely, they exhaustively enumerate all information available at hand to (strategically) optimize some implicit utility function that quantifies

1. NoiseTube, http://www.noisetube.net/index.html.
2. Waze. Outsmarting Traffic Together, http://www.waze.com/.
3. SeeClickFix, https://seeclickfix.com/pages/311-app.html.

the "net value" of their task contributions.

Our starting point in this work is that users are *boundedly rational* agents. This term was used by Herbert A. Simon in [5] to denote that decision makers can rarely know and evaluate all possible outcomes of their decisions with sufficient precision due to constrained memory and processing capacities as well as limited or imperfect information about the decision environment. Since then, extensive experimental evidence has been accumulated *e.g.*, in [6] [7], suggesting that human decisions seek to *satisfice*[4] rather than optimize. In parallel, the research community in cognitive psychology, behavioral economics and marketing science, have come up with cognitive heuristic decision-making models that aim to match this evidence [8] [9].

An attractive feature of cognitive heuristics is that they constitute *descriptive* decision-making models. They capture salient cognitive processes underlying human choices (parsing of alternatives, elimination, satisficing) that persist across very different choice settings. The decision environment of the MCS user will likely be characterized by: a) limited attention and interference due to competition from other simultaneous tasks (high cognitive load); b) limited time to evaluate the proposed tasks and arrive at a decision (time-pressure). Cognitive load and time pressure are both well known to favor the use of heuristics over more complex normative models of human decision-making [6].

Our paper leverages the work on cognitive heuristics to mark a radical departure from standard modeling practice in MCS literature. To the best of our knowledge, it is one of the first attempts to import knowledge and modeling tools from behavioral science to the problem of task recommendation and incentive allocation in MCS. In our modeling approach, presented in section 2, MCS tasks are represented as feature vectors over a finite vector space and users are boundedly rational and heterogeneous with respect to preferences and interests. We capture the bounded rationality aspect through a decision-making model that originates in the field of cognitive psychology, the Fast-and-Frugal-Tree (FFT) model [10] [11]. With FFTs, users rank the task features and sequentially parse them to decide whether to make a contribution or not. Then, the main question for the MCS platform is how to engineer the user choices to motivate valuable contributions to tasks. The *choice engineering* challenge involves properly selecting tasks to recommend to users and offering monetary (or in-kind) rewards for contributions to them within the constraints of the budget allocated to each task.

Our main contributions come in sections 3 and 4. Therein, we address the *joint task recommendation and incentive reward allocation problem* faced by two types of MCS platforms: *nonprofit platforms*, which seek to maximize the aggregate quality of attracted user contributions to MCS tasks, and *for-profit platforms*, which aim at maximizing their revenue out of commissions charged on those contributions. In the first case, we come up with instances of the Generalized Assignment Problem (*e.g.*, see chapter 7 in [12]), whereas the second case gives rise to both linear and non-linear Integer Programs with mixed packing and covering constraints [13]. We discuss the complexity and approximability of these problems and identify algorithmic solutions to them. Finally, we compare these solutions against heuristics that either partially or fully ignore the bounded rationality aspect in the user choices, highlighting the possible performance gains that are feasible when the MCS task offers are systematically optimized for the "right" user decision-making profiles. Notably, our modeling approach readily generalizes to situations, where the MCS platform presents users with multiple task offers rather than a single take-or-leave it offer. Then, the choice of MCS task can be captured by the Discrete Elimination by Aspects (DEBA) model, a lexicographic model suited to multi-attribute choice settings with multiple alternatives [14].

In section 6 we explain how these models can be trained, iterate on their assumptions, and point to their implications for a broader set of application areas beyond MCS. We contrast our work against the existing literature in section 7 before we conclude in section 8.

4. Satisfice is a "portmanteau" word of satisfy and suffice and implies searching through available alternatives until one is deemed acceptable.

2 System model

In our model, there are three types of actors: entities issuing crowdsensing tasks, hereafter called *task issuers*; users owning smart devices who may choose to contribute to these tasks (*task contributors*); and an online MCS *platform* that intervenes between the two sides. The main mission of this platform is to facilitate user contributions to the issued MCS tasks. A mobile app is used for the interaction of task contributors with the MCS platform.

Formally, let $\mathcal{U}$, with $U = |\mathcal{U}|$, be the set of potential task contributors. These are mobile users with smart devices who have registered with the platform and run the respective app on their devices. Let also $\mathcal{M}$, with $|\mathcal{M}| = M$ be the set of crowdsensing tasks that are managed by the platform and $\mathcal{M}_u, u \in \mathcal{U}$ be the subset of tasks that are eligible for contribution by user u. In general, this set of tasks varies with time, depending on the user's location and possibly other contextual information collected by the mobile app. In this work, we assume that user contributions are elicited by the platform upon distinct time epochs and system snapshots (issued tasks, user locations) so that the set of tasks $\mathcal{M}_u$ is fixed for each user.

2.1 Crowdsensing tasks as multi-attribute choices

Each task $m \in \mathcal{M}$ may be represented as a vector of values on a set of L *features*, $\mathbf{f}_m = (f_m^1, \ldots, f_m^L)$. These features may be numerical or categorical and describe different task characteristics such as the reward offered for contributing to it, which may be monetary or in-kind (*e.g.*, a coupon); the physical location, where the task contribution needs to be carried out; the average time or effort to perform the task; the battery/computational burden of the task on the user's device; and the type of the service, *e.g.*, commercial *vs.* community-oriented, that may be facilitated by the task.

Ongoing crowdsensing tasks are advertised to the mobile end users through the platform and the mobile app. One example of such a task notification could be: *"Coffee place m, at distance d_m from your current location, offers a discount voucher of value r_m for taking a couple of nice photos of it. Click*

on the offer to accept it, upload the photos, and get your voucher." Likewise, an example of a notification for a community-oriented task, could be: *"Download the MapNoise app at your smart device, carry out noise measurements and submit them to a noise-mapping tool that aims to improve the quality of life in our municipality. In return, you get a voucher for free coffee or tea from the municipal kiosk this weekend."* The mobile app could present such task notifications one at a time to the users, as "take it or leave it" offers, or simultaneously, inviting them to make a choice between them.

The tasks may demand contributions from one or more end users. Each task m comes up with a budget B_m, which sets an upper bound on what the task-issuing entity is willing to spend on rewarding task contributors. In general, $B_m \geq 0$, *i.e.*, there may be tasks that do not come up with a budget and rely on purely voluntary user contributions.

For simplicity of exposition, we assume in the sequel that the number of important task features for the user choices is $L = 3$: the *task reward*, expressing the user *profit* out of a contribution; the *task location*, which relates directly to the *cost* a user incurs when contributing to a task[5]; and a binary feature, hereafter called *community orientation*, denoting whether this task serves somehow the local community or a purely commercial purpose. This is a strongly behavioral feature that turns out to be decisive for voluntary contributions on behalf of users.

Apparently, these features are indicative and by no means exhaustive. For instance, another feature that may affect the user choice is the battery consumption related to a task, *e.g.*, when this task involves the activation of the mobile device sensors for data collection and transmission. The actual relevance of different features for each user also depends on the time of the day and the user context (work *vs.* leisure time, transportation mode) and, ideally, is captured by the user profiling process (see section 2.2.1), drawing on the learning processes discussed in section 6.1.

2.2 Users as bounded rational agents

The MCS task contributors demonstrate distinct skill sets, related to personal capacities or capabilities of their smart devices, and preferences, related to personal interests and behavioral traits. Hence, contributions by all users do not equally qualify for a given task. In the coffee place example, an amateur or professional photographer would take better photos than someone who does not practice photography at all. Likewise, in the noise mapping example, the quality of noise measurements depends on the sensitivity of the microphone at the user's smartphone. We denote with q_{um} the quality of contribution a user u can make to task m.

On the other hand, user u comes up with her own preferences over each task in $\mathcal{M}_u$, prioritizing/weighing differently each task feature (interchangeably called decision *cue*[6] hereafter). In principle, the way this happens can be *learned* from historical data, that is, data about past choices users have made when tasks were offered to them. Yet, these data can be combined with different modeling hypotheses about the way users decide. *The crucial modeling hypothesis in this work is that users exhibit bounded rationality*. They neither make nor search for optimal choices. They rather activate simple cognitive processes to select among alternatives, which do not necessarily exhaust the information they have at hand about them.

In this work, the bounded rationality hypothesis is captured through decision-making models from the cognitive psychology literature, called *cognitive heuristics*. These are computationally simple models that aim to describe the cognitive processes underlying the actual decision-making process rather than its outcome only [8]. Both theoretical and empirical research in the area of behavioral sciences has shown that the accuracy of cognitive heuristics is comparable to that of more complex and computationally-demanding models such as regression, neural networks, and classification and regression trees (CARTs) [15].

In particular, we model the user decision-making processes after Fast-and-Frugal Trees (FFTs)[7]. FFTs are deterministic *binary* decision trees [10] [11]. Each level of the tree marks the inspection of a cue. The context of the inspection depends on whether the cue is binary or continuous. For binary cues (*e.g.*, community orientation of a task), "1" typically denotes existence of a cue and "0" its absence. Continuous cues, on the other hand, are compared against *acceptability thresholds*: a cue may favor an alternative (*e.g.*, contribute to the task) if it exceeds a threshold value, when it is positively (*e.g.*, MCS task reward) correlated with that alternative; or, it may favor it if it does *not* exceed a threshold value, when it is negatively (*e.g.*, MCS task distance) correlated with that alternative. At every level of the tree, at least one of the two inspection outcomes results in a decision, *i.e.*, at least one of the child nodes is a leaf node (see Fig. 1).

2.2.1 Modeling user choices as FFTs

For given set of decision cues, the FFTs can be parameterized with respect to the order in which these cues are inspected (cue ranking), and the type of cue inspection outcome (positive or negative) that results in choosing an alternative. Distinct user decision-making classes emerge by combining options for these model parameters, so that users of a given class rank and process the decision cues identically.

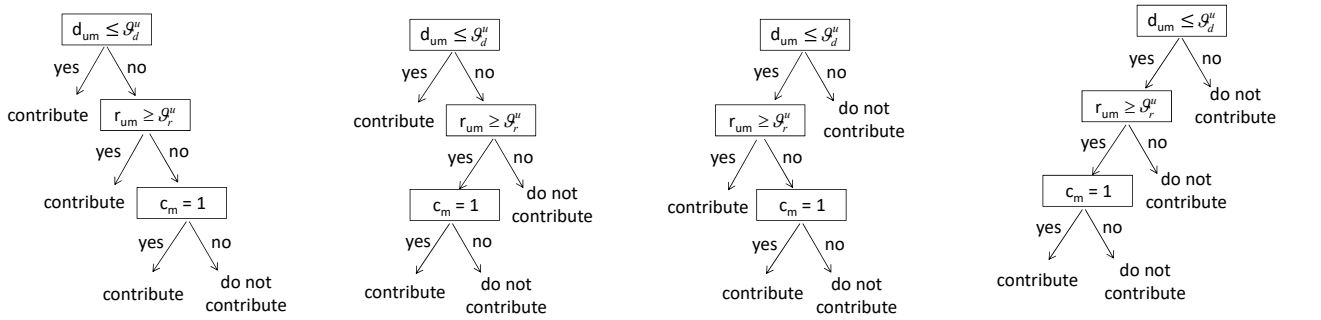
Cue ranking: First, users are grouped according to the order in which they inspect cues. For instance, a user may first inspect the reward offered for contributing to the task, then the distance she has to travel to make a contribution, and, last, whether the task serves somehow the local community. We abbreviate this specific order of inspecting cues as *RDC* out of the initials of the ordered cues: Reward, Distance, Community. There are overall $L! = 6$ different cue rankings in this respect, namely *RDC,RCD,DRC,DCR,CRD,CDR*.

In addition to these six rankings, we need to account for users who are indifferent to one or two of the three cues. Indifference to a cue is simply captured through omission of the respective FFT level. Hence, there are $L(L-1)! = L! = 6$ more rankings of cue pairs (RD,DR,DC,CD,RC,CR)

5. Practically, this cost may correspond to money, time and/or effort spent by the user when traveling to the task location.

6. The terms "feature", "cue", "aspect" are used almost interchangeably in different knowledge areas such as data science, cognitive psychology and consumer research to characterize decision alternatives.

7. They are called *frugal* because they end up processing limited information; and they are called *fast*, because this non-exhaustive processing of available information accelerates the decision-making process [8].



a. FFT type 1 (FFT1 - lenient pectinate) b. FFT type 2 (FFT2 - zigzag) c. FFT type 3 (FFT3 - zigzag) d. FFT type 4 (FFT4 - strict pectinate)

Fig. 1: The four possible FFTs that emerge for users who inspect cues in the order (*distance, reward, community*).

plus $L = 3$ rankings that degenerate to single cues (R,D,C). Hereafter, to keep the model simpler, besides the six three-cue rankings, we consider the two-cue rankings, RD and DR, which capture indifference to whether the MCS task serves the local community or a commercial purpose [16].

Cue inspection outcomes and FFT types: For any of the six rankings involving three cues, users may activate one of the $2^{L-1} = 4$ FFTs that correspond to that ranking. Figure 1 lists these four FFTs for the *RDC* cue ranking. All four FFTs involve checks against the presence of the binary cue (community orientation) and comparisons of the two continuous cue values (task reward, task distance) against the user-specific *acceptability thresholds*, θ_r^u and θ_d^u, respectively. However, in each tree, a choice (*e.g.*, contribute to a task) is reached through different combinations of outcomes in subsets of these checks/comparisons.

Hence, the leftmost and rightmost tree models, the FFTs of type 1 (FFT1) and type 4 (FFT4), are of the pectinate (rake) type: one of the two alternatives is chosen under a strong conjunction rule. Under the FFT4 (strict pectinate), a user chooses to contribute to a task only when *all* three cues result in positive responses ("yes") to the inspection at each level. On the contrary, under the FFT1 (lenient pectinate), a user contributes to the task unless the outcome of *all* three cue inspections is negative ("no"). The other two FFT types, FFT2 and FFT3, are referred to as "zigzag" trees [10] in that positive and negative inspection outcomes alternate in yielding a choice at each level. In FFT2, a choice is reached at first level through a positive inspection outcome, whereas in FFT3 through a negative inspection outcome.

Finally, for users who are indifferent about the community orientation of the task, with cue rankings *DR* and *RD*, there are two pectinate-type FFTs [16].

2.3 MCS platforms as choice engineering tools

The main mission of the platform is to match the issued tasks with mobile users who would be interested in contributing to them. However, the platform does not *assign* tasks to users, for instance, under some prior contractual agreement with them. It rather leverages what it knows about their preferences and the way they make choices (see section 2.2.1) to engineer the *task offers* it issues to them, *i.e.*, the task recommendations coupled with proper rewards for contributions to the tasks. We focus hereafter on platforms that issue take-it-or-leave-it offers for single tasks. In section 5, we discuss ways to expand this work on platforms that issue offers for multiple tasks.

For any given user-task pair, the offered reward is the only decision cue the platform can optimize; the other two are beyond its control. The appropriate reward for a user depends on both her decision-making class (cue ranking, FFT type) and the recommended task. For instance, to users who prioritize based on exerted effort (*e.g.*, distance), the platform may recommend tasks in nearby areas, offering some minimum fixed reward r_{min} predefined and publicly announced by the platform (*default reward*). It could then direct these savings on the task budget to expert users prioritizing tasks according to the offered rewards. Likewise, the platform needs smaller rewards to attract contributions by users of lenient- than by users of strict-pectinate FFT type.

2.3.1 Determining user-specific task rewards

A contribution by user u to task m comes with a positive value q_{um} that corresponds to the quality of the contribution that u can make to m; and a cost r_{um} relating to the reward that should be offered to u to ensure her contribution. Given each user's u decision-making class, a critical requirement for the platform is to determine the *minimum* reward that suffices to induce a contribution of u to task m.

The set of tasks $\mathcal{M}_u$ in the neighborhood of user u may be split into $2^{L-1} = 4$ user-specific *task groups*:

$$\begin{aligned}
\mathcal{M}_{u,1} &= \{m \in \mathcal{M}_u : d_{um} \leq \theta_u^d, c_m = 1\} \\
\mathcal{M}_{u,2} &= \{m \in \mathcal{M}_u : d_{um} \leq \theta_u^d, c_m = 0\} \\
\mathcal{M}_{u,3} &= \{m \in \mathcal{M}_u : d_{um} \geq \theta_u^d, c_m = 1\} \\
\mathcal{M}_{u,4} &= \{m \in \mathcal{M}_u : d_{um} \geq \theta_u^d, c_m = 0\}
\end{aligned}$$

where the indicator variable $c_m = 1$ when a task serves the local community and $c_m = 0$, otherwise. The minimum rewards that induce positive responses to task offers can then be computed per task group and user decision-making class, see Table 1. We can make the following remarks:

- The two relevant reward values for a given user are her reward acceptability threshold, θ_u^r, and the default platform reward, r_{min}. For all platform users, these reward values are $O(|U|)$ since the reward acceptability thresholds are, in principle, user-specific.
- The minimum rewards are similar for all pairs of user decision-making classes that inspect the same

TABLE 1: Minimum rewards that can induce a positive response of user u to a task offer $m \in \mathcal{M}_u$ as a function of her decision-making class, FFT type, and the task group. For the RD and DR cue rankings, FFT1 and FFT4 point to the lenient- and strict-pectinate trees in [16]

Task group	Cue rank / FFT type	DCR, DRC				RDC, RCD				CRD, CDR				RD, DR	
		FFT1	FFT2	FFT3	FFT4	FFT1	FFT2	FFT3	FFT4	FFT1	FFT2	FFT3	FFT4	FFT1	FFT4
$\{d_{um} \leq \theta^d_u, c_m = 1\}$		r_{min}	r_{min}	r_{min}	θ^u_r	r_{min}	r_{min}	θ^u_r	θ^u_r	r_{min}	r_{min}	r_{min}	θ^u_r	r_{min}	θ^u_r
$\{d_{um} \leq \theta^d_u, c_m = 0\}$		r_{min}	r_{min}	θ^u_r	–	r_{min}	θ^u_r	θ^u_r	–	r_{min}	θ^u_r	–	–		
$\{d_{um} \geq \theta^d_u, c_m = 1\}$		r_{min}	θ^u_r	–	–	r_{min}	θ^u_r	θ^u_r	–	r_{min}	r_{min}	θ^u_r	–	θ^u_r	–
$\{d_{um} \geq \theta^d_u, c_m = 0\}$		θ^u_r	–	–	–	θ^u_r	θ^u_r	–	–	θ^u_r	–	–	–		

cue first. In other words, the rewards vary with task group and FFT type but they do not depend on the precise order in which the decision cues are inspected; only the first cue matters.

- The rewards across the four task groups exhibit the same pattern under all six cue rankings for FFT type 1 and FFT type 4 users.
- By definition, users activating the strict-pectinate type of FFT (FFT4) are the least flexible since they will only accept offers for nearby tasks that benefit the community and offer rewards at least as high as the reward acceptability threshold θ^r_u. The *number* of tasks that are candidate for offer to each user depends mainly on the FFT type and less on the cue ranking.

All in all, the originally 28 decision-making classes can be merged into 10 *reward allocation classes*. If we denote each such class by the pair (*cue ranking*, *FFT type*), the ten classes are $(*, FFT1)$, $(*, FFT4)$, where * is a wildcard for any of the six cue orderings involving three cues; the six combinations of type (a, b), where $a \in \{(DCR \cup DRC), (RCD \cup RDC), (CRD \cup CDR)\}$ and $b \in \{FFT2, FFT3\}$; and the two classes $((RD, DR), c)$, where $c \in (FFT1, FFT4)$. Users within each of these classes could be treated in similar manner by the platform in terms of the minimum reward that should be proposed to them for any given task in $\mathcal{M}_u$.

We optimize the task offers for two types of platforms. Non-profit platforms typically prioritize the quality of attracted user contributions. This could be the case with platforms administered and funded by a public institution (*e.g.*, municipality) or a non-profit entity. The non-profit orientation could also be an intermediate step in the platform's growth strategy, while trying to scale up and set up the two sides of the market, the market supply (task issuers) and demand (task contributors). For-profit platforms, on the other hand, aim at maximizing their revenue, which results from commissions out of each task contribution they manage to attract. The commissions may either be fixed or in proportion to the user rewards. In either case, the allocation of rewards to the mobile users is carried out within the limits of task budgets, as defined by the task issuing entities.

3 Task offers in nonprofit MCS platforms

3.1 Optimizing the aggregate quality of contributions

If $\mathbf{x} = (x_{um} : u \in \mathcal{U}, m \in \mathcal{M}_u)$ are binary decision variables, with $x_{um} = 1$ if user u is made an offer for task m, and $x_{um} = 0$ otherwise, the optimization problem (P1) faced by the MCS platform can be written as follows:

$$\max_{\mathbf{x}} \quad \sum_{u \in \mathcal{U}} \sum_{m \in \mathcal{M}_u} q_{um} x_{um}, \tag{1}$$

$$s.t. \quad \sum_{u: m \in \mathcal{M}_u} r_{um} x_{um} \leq B_m \quad \forall m \in \mathcal{M} \tag{2}$$

$$\sum_{m \in \mathcal{M}_u} x_{um} \leq 1 \quad \forall u \in \mathcal{U} \qquad (P1) \tag{3}$$

$$x_{um} \in \{0, 1\} \;\; u \in \mathcal{U}, m \in \mathcal{M}_u \,. \tag{4}$$

where the rewards r_{um} offered to users are determined as in section 2.3.1. The problem (P1) is an instance of the maximum Generalized Assignment Problem (GAP), referred to as LEGAP in [12]. With regard to the standard GAP typology, MCS tasks in (P1) correspond to bins, task budgets to bin capacities, user contributions to items, qualities of user contributions to item profits, and rewards offered for task contributions to item sizes. Equation (2), in particular, reflects the per task budget constraints on the rewards that can be offered to users as incentives for their contributions. It is trivial to show, yet worth codifying as proposition for future reference, that:

Proposition 1. *The MCS platform cannot further increase the aggregate quality of contributions in (1) by offering rewards beyond the minimum ones reported in Table 1.*

The best approximation ratio for the GAP in the form of (P1)[8], $\frac{e}{e-1} + \epsilon$ for $\epsilon > 0$, is achieved by the LP-based algorithm in [17]. Comparable approximation guarantees are also achievable with the combinatorial algorithm proposed in [18], which reduces the problem to iteratively solving 0-1 Knapsack Problems (KPs) for each task. The authors in [18] show that any α-approximation algorithm A for the KP with running time $f(U)$ can be transformed into an $(1 + \alpha)$-approximation algorithm for GAP that runs at $O(Mf(U) + MU)$ time.

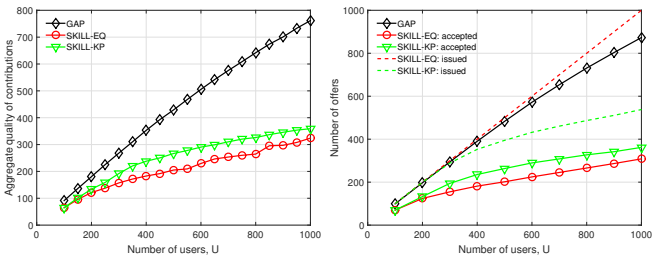
3.2 Evaluation

In this section, we evaluate the achievable performance gain when the MCS platform explicitly accounts for the decision-making strategies of end users.

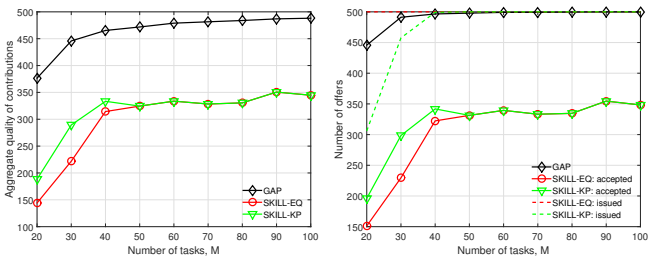
3.2.1 Methodology

We simulate several instances of the joint task recommendation and reward allocation problem. Each problem instance specifies the number of MCS tasks, M, and their spatial distribution across a rectangular area of RxR m^2; the number

8. The problem variant that emerges when (3) holds strictly as equality has also been studied in the literature, *e.g.*, see [12].



(a) $M = 25$, variable number of users



(b) $U = 500$, variable number of tasks

Fig. 2: Approximately optimal task offers *vs.* alternative heuristic rules: $R = 1000$, $\theta_r^u \sim unif(0.5, 3.5)$, $\theta_d^u \sim unif(170, 1000)$, $r_{min} = 0.25$.

of involved users, U, and their positions within the same area; and the decision-making profiles of the users, *i.e.*, their decision-making classes (cue ranking, type of FFT) with the relevant thresholds $\{\theta_r^u\}$ and $\{\theta_d^u\}$. The task and user locations, as well as the decision-making profiles of users (FFT type, threshold values) are randomly chosen, in line with different statistical distributions. For each problem instance, we solve the optimization problem (P1) for the maximum aggregate quality of contributions. Unless otherwise stated, the task budgets are uniformly set to $B_m = 25, \forall m \in \mathcal{M}$.

Comparison references for nonprofit platforms: This approximately optimal solution of (P1), marked as GAP in the plots of this section, is compared with two heuristics for prescribing task offers to users. The first one issues recommendations to each user u for the task $m \in \mathcal{M}_u$ that she is most skilled for, *i.e.*,

$$x_{um} = 1, \quad m = \arg\max_{l \in \mathcal{M}_u} q_{ul}, \quad u \in \mathcal{U}\,. \tag{5}$$

and then splits the budget of each task equally among those users who get an offer for it, *i.e.*,

$$r_{um} = \frac{B_m}{K}, \quad K = |\{u : x_{um} = 1\}| \tag{6}$$

The second heuristic uses the same rule (5) for identifying *candidate* contributors for each task. However, it then solves a 0-1 KP, where the MCS task is the bin and the users are the items to pack in it, taking into account their skills $\{q_{um}\}$ and their reward acceptability thresholds $\{\theta_r^u\}$.

In what follows, we refer to the two heuristics with the abbreviations SKILL-EQ and SKILL-KP, respectively. Notably, the heuristics leverage different amounts of user profiling information. SKILL-EQ only relies on the skills of users (and their devices) and myopically chooses tasks to offer inline with th0se skills. On the other hand, SKILL-KP also exploits information about what users consider acceptable as reward for a task offer and optimizes tasks offers on the basis of this coarse information. Comparing GAP with these heuristics, we can have a better understanding of what we gain by undertaking the analysis in section 2.3.1 and then drawing on it to optimize task offers in section 3.1.

3.2.2 Results

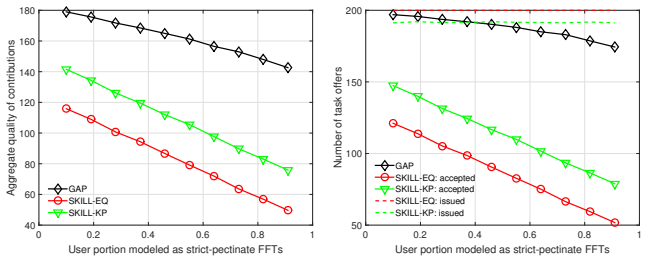
General performance trends: Figure 2a plots the number of task offers, both those issued by the MCS platform and those accepted by the end users, and the resulting aggregate quality of user contributions to MCS tasks under a broad range of U/M ratios (user density per MCS task). In these experiments, the MCS users are equally split between the ten reward allocation classes discussed in section 2.3.1 and approximately half the tasks have a community orientation.

In principle, as the number of the platform users increases for given M (top row), the platform can direct task offers to more skilled users. On the other hand, in light of the budget constraints, it is far harder to come up with "good" offers for all users. GAP leverages the enhanced profiling information about users and tasks in section 2.3.1 to optimize task offers; all its offers are tailored to the user-specific preferences and task types and induce contributions by users. However, as the number of users scales up, beyond $U = 400$ when $M = 25$, the portion of users who get offers gracefully drops. On the contrary, the SKILL-EQ heuristic always make offers to *all* users in the platform but, since it only accounts for the user skills, most of them are turned down by users. In fact, the scheme does not scale to high U/M ratios: the finite budget is split into equal but very small rewards that cannot motivate user contributions.

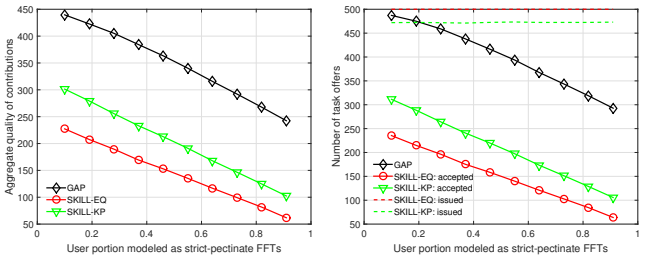
Finally, the SKILL-KP scheme is at least as good as SKILL-EQ at low user densities and almost always outperforms it at high user densities. The scheme is much more selective when making offers, ensuring that the issued offers come up with rewards equal to the users' reward acceptability threshold values. However, this can be wasteful for two reasons, as can be inferred from Table 1: either because a smaller offer, *e.g.*, r_{min}, would suffice to ensure a user contribution to the specific task or because the task is such that the user would never consider it for a contribution, irrespective of the offered reward (blank entries in Table 1).

Impact of the user decision-making classes' mix: With up to three decision cues, there are already 28 user decision-making classes and 10 reward allocation classes; with L cues the user decision-making classes are $O(L!2^L)$ and the possible partitions of MCS users to these decision-making classes are $O(L!2^L)^U$. In Fig. 3 we attempt to get an idea of how the mix of user decision-making classes impacts the effectiveness of our scheme by considering two symmetric scenarios, where a varying majority of users is modeled by strict(lenient)-pectinate FFTs models and the rest are spread equally among the remaining classes. In a sense, these two scenarios represent opposite extremes, where the MCS platform has to cope with least (most) flexible MCS users, respectively.

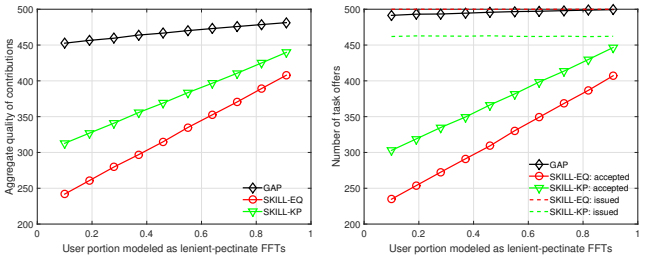
Fig. 3 implies that the GAP scheme exhibits higher robustness than its competitor heuristics to the variations in the user decision-making class mix. When strict-pectinate users dominate the mix (Fig. 3a, b), the GAP performance degradation is more graceful (sublinear) and its relative



(a) U=200, M=20, strict-pectinates



(b) U=500, M=30, variable strict-pectinates



(c) U=500, M=30, variable lenient-pectinates

Fig. 3: Aggregate task contribution quality (left) and accepted offers (right) under the approximately optimal and heuristic alternatives *vs.* mix of strict- and lenient-pectinate users, R = 1000, $\theta_r^u \sim \mathcal{U}(0.5, 3.5)$, $\theta_d^u \sim \mathcal{U}(170, 1000)$.

performance gain improves as this demanding class of users grows. When lenient-pectinate users dominate (Fig. 3c), and despite the linear performance improvement of the heuristic rules, GAP retains its performance advantage by sublinearly increasing the quality of its attracted user contributions.

Impact of the reward acceptability threshold: Higher spread of the reward threshold values θ_r^u implies more users with higher reward demands for contributing to a task. Figure 4 plots the aggregate quality of user contributions against increasing values of $r_{max} = \max_u \theta_r^u$, under U/M ratios in the range [10,33]. The users in Figs. 4a), b), d) are equally split between the reward allocation classes, whereas in Fig. 4c strict-pectinate users dominate the mix.

Overall, Fig. 4 suggests that the performance advantage of GAP is not sensitive to the minimum reward requirements of MCS users. It manages the task budgets far more efficiently so that its performance gain over the second best option remains the same or increases in the presence of more "expensive" MCS users (high r_{max} values).

Decomposition of the performance gain: In all scenarios plotted in Figs. 2-4, and in many more not shown here due to lack of space, GAP significantly outperforms alternative heuristics for managing task offers. The gain in terms of aggregate quality of user contributions ranges from 20% (Fig. 4c) up to more than 300% (Fig. 2b), when compared to the second best alternative.

In this subsection, we want to assess how much of this performance gain is due to the user profiling effort in sections 2.2.1 and 2.3.1 and how much relates to the optimization approach in section 3.1 *per se*. Hence, we compare GAP with two purposefully designed alternatives. The first one solves (P1) to match task offers with users but rewards user u in proportion to her skills for task m:

$$r_{um} = r_{min} + q_{um}(r_{max} - r_{min}) \tag{7}$$

The second scheme uses the minimum rewards computed in Table 1 but matches task offers with users in two steps. First, it provisionally "assigns" users to tasks they are most skilled for. Then, it works independently with each task and if the task budget suffices to issue offers to all users, the platform does so; if not, a 0-1 KP instance is solved to select the most valued (*i.e.*, skilled) users.

Hence, the first scheme (OPT-PROP) optimizes the task recommendations, but uses a heuristic rule to determine the offered rewards. On the contrary, the second scheme (SKILL-OPT) uses a heuristic rule to determine the tasks recommended to the users but takes into account the user profiling work to optimize the rewards offered to them.

Figure 5 shows that the user profiling effort has much more impact on performance than the optimization process. Nevertheless both schemes exhibit a better performance than the comparison heuristics SKILL-*, and their combination exhibits the best performance of all, as we expected.

4 Task Offers in For-Profit MCS Platforms

For-profit MCS platforms primarily aim at maximizing their revenue. This comes out of a commission fee charged each time a user contribution is made. Charging commission fees per transaction facilitated by the platform is a more general practice in online and sharing economy platforms. Consider, for example, Airbnb[9] or the most popular ride-sharing applications such as Uber or Lyft[10].

We consider two possibilities for these commission fees. In the first case (*fixed commission*), the fee is a fixed amount h_f for each delivered user-to-task contribution. In the second case (*fixed commission rate*), the fee is a fixed ratio h_r of the reward allocated to a user. Hence, it varies across user contributions. This case is reminiscent of the way Airbnb charges commissions upon accommodation rentals.

4.1 Fixed commission fee

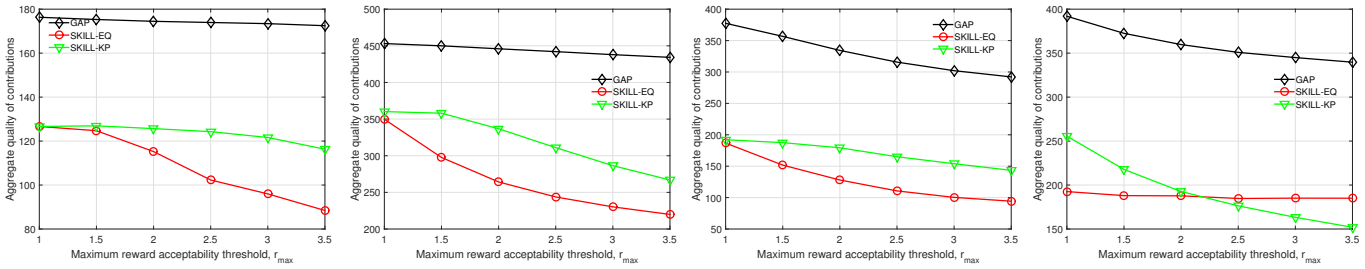
Since the commission fee is fixed, the platform revenue is proportional to the absolute number of attracted user contributions to MCS tasks. The optimization problem (P2) faced by the MCS platform can be stated as follows.

$$\max_{\mathbf{x}} \quad \sum_{u \in \mathcal{U}} \sum_{m \in \mathcal{M}_u} x_{um}\,, \tag{8}$$

$$s.t. \quad (2), (3), (4) \qquad (P2)$$

$$\sum_{u: m \in \mathcal{M}_u} q_{um} x_{um} \geq Q_m \quad \forall m \in \mathcal{M} \tag{9}$$

9. http://www.airbnb.com
10. http://www.uber.com, http://www.lyft.com.



(a) U=200, M=20, uniform mix (b) U=500, M=25, uniform mix (c) U=500, M=25, 65% strict pectinates (d) U=500, M=15, uniform mix

Fig. 4: Sensitivity of aggregate quality of task contributions to the user reward acceptability threshold: R = 1000 $\theta_r^u \sim \mathcal{U}(1, r_{max})$, $\theta_d^u \sim \mathcal{U}(170, 1000)$.

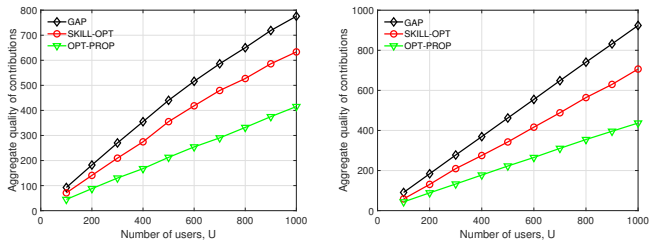


Fig. 5: Decomposition of the performance gain: (a) $M = 25$, $B_m = 25,\ m \in \mathcal{M}$, (b) $M = 50$, $B_m = 1.5 \cdot U/M,\ m \in \mathcal{M}$.

TABLE 2: Average running times in seconds (30 samples for each (U,M) pair) of the default branch-and-bound MILP solver in MATLAB `intlinprog` with an AMD ryzen 3 2200U CPU with Radeon Vega Mobile Gfx 2.50GHz and 4GB RAM. Outliers are excluded from the average (with probability ∼0.01 a problem instance was infeasible or its running time exceeded 60 sec.)

		Number of tasks				
		10	20	30	40	50
Number of users	100	0.11	0.11	0.58	0.50	0.94
	200	0.20	0.56	0.83	0.42	0.64
	300	0.49	3.08	0.51	4.98	3.92
	400	0.55	1.07	2.47	4.87	9.45
	500	0.62	2.83	3.90	6.29	7.33
	600	1.57	2.58	4.38	8.14	6.90

In constraint (9), Q_m is a minimum threshold imposed on the quality of contributions that will be attracted for task m. Different tasks may have stricter or looser requirements in this respect. For example, a photo-shooting task comes with a higher Q_m value, requiring at least some photos by (amateur) photographers with high resolution smartphone cameras. We discuss the practical issues regarding the calibration of such values in section 6.

Problem (P2) belongs to the broader family of ILP problems with both covering- and knapsack/packing-type constraints. Kolliopoulos and Young in [19] provide a bicriteria-approximation algorithm for the minimization version of the most generic form of the problem $\{\min c^T x : x \in \mathcal{Z}_+^n, Ax \geq \alpha,\ Bx \leq b,\ x \leq d\}$ including covering ($Ax \geq \alpha$), packing ($Bx \leq b$), and multiplicity constraints ($x \leq d$), setting an upper bound on the integer decision variables. Their algorithm yields solutions that are $O(ln(D)/\epsilon^2)$ of the optimal ones, where D stands for the number of packing constraints (number of rows in A) and $\epsilon \in (0, 1]$, while imposing upper bounds on the violation of the covering constraints, $Bx \leq (1+\epsilon)b+\beta$, with $\beta_i = \sum_j B_{ij}$. In our case, we typically cope with "small" problems featuring tens of tasks and hundreds of users. For the needs of the evaluation in section 4.4, we use mainstream ILP solvers that return exact solutions in tens of seconds worst-case (see Table 2).

4.2 Fixed commission rate

When the platform's commission is proportional to the rewards of task contributors, Proposition 1 does not, generally, hold. The platform could still use these minimum rewards as a starting point for finding *any* feasible solution x_0 under constraints (2), (3), (4) and (9). It could then *a posteriori* inflate the rewards to the task contributors in x_0, to exhaust the MCS task budgets and maximize its profits. However, the reward inflation process could result in asymmetrically high rewards to individual contributors. This is not good practice for an MCS platform in the long term.

To come up with more plausible solutions in this case, let $\rho_{um} \geq r_{um}$ be the rewards offered by the platform under fixed commission rate, $h_r \in [0, 1]$, and let r_{max} be a maximum reward value that is fixed for the MCS platform. The total revenue of the platform will be equal to $h_r \sum_{u \in \mathcal{U}} \sum_{m \in \mathcal{M}_u} \rho_{um} x_{um}$, but since h_r is a constant, the optimization problem faced by the platform is given by

$$\max_{\mathbf{x},\boldsymbol{\rho}} \quad \sum_{u \in \mathcal{U}} \sum_{m \in \mathcal{M}_u} \rho_{um} x_{um} \qquad (P3)$$

$$s.t. \quad \sum_{u: m \in \mathcal{M}_u} \rho_{um} x_{um} \leq B_m \quad \forall m \in \mathcal{M} \qquad (10)$$

$$r_{um} \leq \rho_{um} \leq r_{max} \quad \forall m \in \mathcal{M}, \forall u \in \mathcal{U} \qquad (11)$$

$$(3), (4), (9)$$

The problem (P3) is an instance of bilinear programming [20], with mixed integer and real variables that make it computationally hard. In practice, problem sizes of interest to our case can be solved very efficiently with an integer programming solver, after applying linearization techniques to the product of decision variables (*e.g.*, [21]).

Observe that a theoretical upper bound for the optimum of (P3) is the minimum between $U \cdot r_{max}$ and $\sum_m B_m$.

4.3 Comparison of problems (P2) and (P3)

If the minimum (r_{min}) and the maximum (r_{max}) reward values are close to each other, it turns out that the optimal solution of (P2), which makes payments according to Table 1, approximates well the optimal solution of (P3).

Formally, let $P2(\mathbf{x}) = \sum_{u,m} x_{um}$ be the objective function of (P2) and $P3(\rho, \mathbf{x}) = \sum_{u,m} \rho_{um} x_{um}$ be the objective of (P3), with $\sum_{u,m}$ abbreviating the double summation operator $\sum_{u \in \mathcal{U}} \sum_{m \in \mathcal{M}_u}$. If $\hat{\mathbf{x}}$ denotes an optimal solution to (P2), $(\rho^*, \mathbf{x}^*)$ denotes an optimal solution to (P3) and the set of minimum rewards $\mathbf{r}$ is given by Table 1, we can prove that

Theorem 1. *If $r_{max} \leq c \cdot r_{min}$, where $c \geq 1$, then*

$$\frac{1}{c} P3(\rho^*, \mathbf{x}^*) \leq P3(\mathbf{r}, \hat{\mathbf{x}}) \leq P3(\rho^*, \mathbf{x}^*). \tag{12}$$

Proof. First of all it is easy to check that $(\mathbf{r}, \hat{\mathbf{x}})$ is a feasible solution to (P3). The second inequality in (12) is straightforward since $(\rho^*, \mathbf{x}^*)$ is an optimal solution to (P3).

Regarding the first inequality in (12) we have that

$$P3(\rho^*, \mathbf{x}^*) = \sum_{u,m} \rho^*_{um} x^*_{um} \leq \sum_{u,m} r_{max} x^*_{um} =$$

$$r_{max} \sum_{u,m} x^*_{um} \leq c \cdot r_{min} \sum_{u,m} x^*_{um} \leq c \cdot r_{min} \sum_{u,m} \hat{x}_{um},$$

where the last holds since $\hat{\mathbf{x}}$ is an optimal solution to (P2).

Now since $\forall u, m \; r_{min} \leq r_{um}$, we have

$$c \cdot r_{min} \sum_{u,m} x^*_{um} \leq c \cdot \sum_{u,m} r_{um} x^*_{um} = c \cdot P3(\mathbf{r}, \mathbf{x}^*).$$

□

It suffices to multiply all parts of the inequalities in (12) by h_r to obtain that the approximation in Theorem 1 also holds for the revenue of the platform

Corollary 1. *If $r_{max} \leq c \cdot r_{min}$, where $c \geq 1$, then*

$$\frac{h_r}{c} P3(\rho^*, \mathbf{x}^*) \leq h_r \cdot P3(\mathbf{r}, \hat{\mathbf{x}}) \leq h_r \cdot P3(\rho^*, \mathbf{x}^*).$$

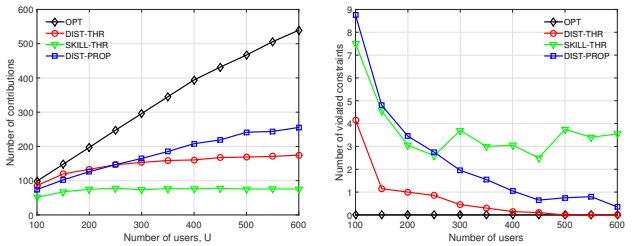
Note that if an optimal solution to (P2) is tight with respect to the budget constraints, then it is optimal to (P3) as well (regardless of the r_{min} and r_{max} values).

4.4 Evaluation

4.4.1 Methodology

Methodologically, the setting in these experiments is similar to that in section 3.2.1. In particular, the reward thresholds of MCS users follow the uniform distribution over [0.5,3] and their skills take values in [0.1,1] according to the normal distribution $\mathcal{N}(0.55, 0.15)$. Finally, each user is assigned to one of the decision-making classes in Table 1 independently and uniformly at random. For each problem instance, we solve the optimization problems (P2) and (P3) for the maximum platform revenue, see sections 4.1 and 4.2, and we compare their solutions against those of alternative schemes. Each point in the plots is the average of 40-50 simulation runs.

Comparison references for for-profit platforms: We consider three heuristics. The first one, *DIST-PROP*, offers each user her nearest task, to ensure that she will not reject the offer due to the task distance. It then distributes the task budget in proportion to the skills of each user, to favor contributions of higher quality to the task.



(a) Attracted contributions (b) Violated quality constraints

Fig. 6: Attracted task contributions and number of violated constraints *vs.* number of users in for-profit platforms charging fixed commission fees: $M = 20$, $Q_m = 1.5$, $B_m = 15$.

The other two heuristics offer rewards that equal users' reward acceptability thresholds $\{\theta^r_u\}$ but differentiate as to how they determine the task recommendations to users. The *DIST-THR* heuristic parses tasks sequentially and greedily chooses users, in order of increasing distance from the task, until either the task quality threshold is met or the task budget is exhausted. Any remaining users are assigned to their nearest task. On the contrary, the *SKILL-THR* heuristic first maps users to the tasks they are most skilled for. Then, it seeks to maximize the number of attracted contributions by paying as many of them as possible in order of increasing θ^u_r, *i.e.*, starting from the cheapest ones.

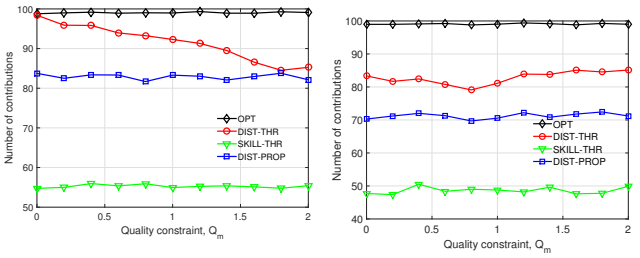
Note that the feasibility of problems (P2) and (P3) is not guaranteed. Hence, together with the revenues achieved by the optimal solution (OPT) and its comparison references, we also report the number of infeasible problem instances and quality constraints the four alternatives fail to satisfy.

4.4.2 Numerical results

We highlight the main performance properties of our solution for platforms charging fixed commission fees through focused experiments. We discuss platforms charging fixed commission rates in section 4.4.3.

Impact of the number of platform's users: As expected, the MCS platform revenue is non-decreasing with the user supply under all four approaches to the joint task recommendation and reward allocation problem. However, it can also be seen in Fig. 6a that with the optimal solution to the formulation (P2) the revenue scales far more aggressively than with its alternatives, linearly up to 400 users and sublinearly for more users. OPT directs efficiently the MCS task offers and satisfies all quality constraints on the attracted contributions' quality (Fig. 6b). On the contrary, as the user supply grows, the three alternatives are presented with higher flexibility to satisfy the task quality constraints Q_m (at least, the DIST-THR and DIST-PROP heuristics) but cannot scale up the number of attracted contributions and, hence, the platform revenue.

Impact of the quality constraint: As a general remark out of Fig. 7, the number of contributions that two out the three heuristic solutions attract appear to be insensitive to the task quality constraint size. What changes, as Q_m ranges in [0,2] is that the heuristic schemes fail to satisfy an increasing number of the task quality constraints in (9), as shown in Table 3. The task budget level, as expected, has a more visible impact on the attracted contributions, deter-



(a) $U = 100, B_m = 20$ (b) $U = 100, B_m = 10$

Fig. 7: Attracted task contributions *vs.* quality constraint, Q_m, in for-profit platforms with fixed commission: $M = 25$.

TABLE 3: Infeasible instances and violated constraints of the *best* out of the three heuristic solutions to (P2) as a function of the quality constraint, Q_m: U=100, M=25, B_m=6.8

Q_m	0.8	1.2	1.4	2	2.1	2.2	2.4
Infeasible instances	0%	0%	<1%	<1%	29%	66%	>90%
Violated constraints	2.3	6.1	11.4	17.5	20.6	20.8	21.7

mining what is the achievable platform revenue. Noticeable is the performance variation of the DIST-THR heuristic for $B_m = 20$ in Fig. 7a. It matches the optimal solution when the quality constraint is inactive and strongly outperforms DIST-PROP for low Q_m values since its offered rewards render the task offers acceptable by more users.

At the same time, the ILP solver comes up with feasible solutions to (P2) almost always for Q_m up to 2. It does so even under tighter task budget values, deepening the performance gap from its alternatives. Overall, OPT turns out to be far more resilient to both the covering (*i.e.*, task quality) and packing (*i.e.*, task budget) constraints that apply to the platform revenue maximization.

Impact of task budget: Finally, we plot the revenue attracted by the four alternatives as a function of the task budget. We set $Q_m = 1.5$, to ensure few constraint violations, even under low task budgets (see Table 3).

In all cases, Fig. 8 suggests that formulating the problem as in (P2) and solving it optimally proves to be far superior to the way the other heuristics manage the per-task budgets. OPT can elicit contributions from almost all users even at low budget values, whereas the heuristics need far higher budgets to scale up contributions. In the best case (DIST-THR heuristic, Fig. 8a) the achievable revenue can match that under the optimal solution but for task budgets 3-4 times higher. Typically, even at those budget levels, the best achievable revenue with the heuristics levels off approximately 10% below the optimal values (see Figs. 8b-d) .

4.4.3 Nonprofit vs. for-profit platforms

Finally, we compare the three types of platforms with each other. To do this we find the optimal solutions to problems (P1), (P2) and (P3), and then compute the aggregate quality, the number of contributions, and the sum of payments achieved by each one of them. Our purpose is to determine to what extent the aggregate task quality and revenue objectives are aligned or stand in conflict with each other. Particular care is taken to make platforms charging fixed commission rates comparable to the rest, with respect to the sum of payments. Namely, we inflate the rewards allocated to the task offers made by the other two types of platforms until either the payments reach the upper limit of r_{max} or a task budget is exhausted.

Table 4 reports the worst case scores of the three platforms in each of the three objectives. For instance, the nonprofit platform (P1) achieves 97% (75%) of the revenue of the for-profit platform that charges fixed commission fees (commission rates, respectively).

	Var $B_m \in [5, 25]$			Var $Q_m \in [0.5, 2.5]$		
	P1	P2	P3	P1	P2	P3
Aggr. quality	100%	78%	57%	100%	84%	60%
No contributions	97%	100%	75%	98%	100%	80%
Sum rewards	75%	80%	100%	94%	94%	100%

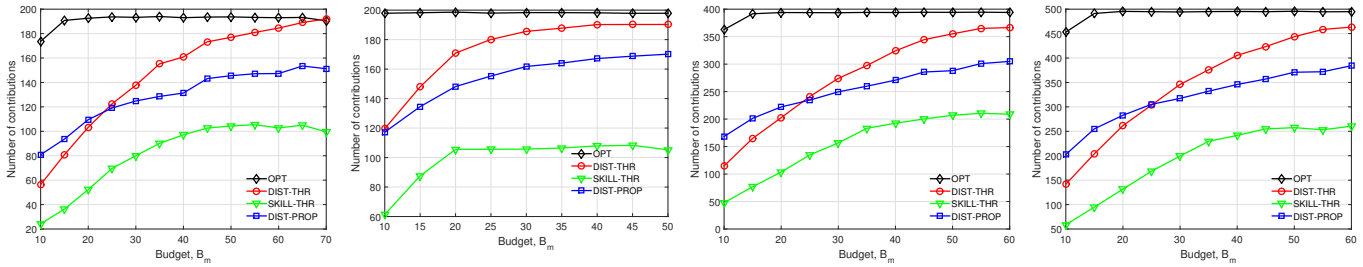
TABLE 4: Worst-case ratio of the scores achieved by the three types of platforms to the optimum of each objective.

While optimizing for the aggregate quality of contributions, the nonprofit platform (P1) also approximates well the revenue achieved by the for-profit platforms (worst-case approximations 97%, 75%, 98%, 94%). In contrast the platform that maximizes the sum of payments (P3) gives a poor aggregate quality (57% and 60% in worst case, 72% in best case -best case table omitted-). The two for-profit platforms (P2, P3) both approximate well each others' objectives (75%, 80%, 80% and 94%). An interesting open question and direction for future work concerns the deduction of more general theoretical results as to how these three objectives and the respective MCS platforms compare with each other.

5 Platforms Issuing Multiple Task Offers

Now, assume that the mobile app offers two alternatives (MCS tasks) m_1, m_2 to each user u, specified by the reward-distance pairs (r_{um_1}, d_{um_1}) and (r_{um_2}, d_{um_2}). The user may choose to contribute to one of the two tasks recommended by the app or decline both offers.

Since FFTs model binary choices, they cannot capture the decision process in this case. The most relevant heuristic, which sequentially iterates over lexicographically ordered cues to single out one out of many (≥ 2) alternatives, is the Deterministic Elimination By Aspects (DEBA) [14]. DEBA ranks cues in order of decreasing importance (x_1, x_2,...). With discrete cues, the alternatives are readily codified as ordered sequences of ones and zeros, hereafter called *DEBA encodings*, depending on whether they possess or not a binary attribute. DEBA then inspects the value of all alternatives on x_1 and eliminates those with $x_1 = 0$. The process is repeated when parsing the second and remaining cues until a single alternative remains. If more than one alternatives are left after all cues are inspected, or if there is a cue eliminating all remaining alternatives, a choice is made randomly among the currently surviving ones. Continuous cues, on the other hand, can be discretized by comparing them with the threshold acceptability values. An alternative that scores favorably in the cue, *e.g.*, a task with reward exceeding θ_r^u or a task at distance smaller that θ_d^u, assumes an ace in the respective position of its DEBA encoding.



(a) $M = 10, U = 200$ (b) $M = 25, U = 200$ (c) $M = 20, U = 400$ (d) $M = 25, U = 500$

Fig. 8: Attracted task contributions under four solutions to the optimization problem (P2) *vs.* task budget: $Q_m = 1.5$.

TABLE 5: Minimum rewards that have to be offered to user u to induce a positive response to a task offer as a function of her DEBA decision-making class and the task group.

Cue ranking / Task group	DCR	DRC	RDC	RCD	CRD	CDR	RD	DR
$\{m \in \mathcal{M}_u : d_{um} \leq \theta^d_u, c_m = 1\}$	$r_{min}(110)$	θ^u_r (111)	θ^u_r (111)	θ^u_r (111)	$r_{min}(101)$	$r_{min}(110)$	θ^u_r (11)	θ^u_r (11)
$\{m \in \mathcal{M}_u : d_{um} \leq \theta^d_u, c_m = 0\}$	–	θ^u_r (110)	θ^u_r (110)	θ^u_r (101)	–	–	θ^u_r (11)	θ^u_r (11)
$\{m \in \mathcal{M}_u : d_{um} \geq \theta^d_u, c_m = 1\}$	–	–	θ^u_r (101)	θ^u_r (110)	r_{min} (100)	r_{min} (101)	θ^u_r (10)	–
$\{m \in \mathcal{M}_u : d_{um} \geq \theta^d_u, c_m = 0\}$	–	–	θ^u_r (100)	θ^u_r (100)	–	–	θ^u_r (10)	–
DEBA encoding of virtual choice	100	100	001	010	010	001	01	10

With DEBA, the option of not contributing to either of the two tasks can be conveniently modeled as a third virtual task. Then Table 5 is the counterpart of Table 1 for paired-task offers under the DEBA heuristic. It lists the user decision-making classes and the minimum rewards, when such exist, that can induce user contributions to the four task types $\mathcal{M}_1$-$\mathcal{M}_4$ in (1). It also provides the DEBA encodings of the resulting task offers and the virtual alternative ("do not contribute"). The eight user decision-making classes stand in one-to-one correspondence with the eight possible cue rankings, which reduce to four different reward allocation classes ({DCR}, {DRC,DR}, {RDC,RCD,RD}, {CRD,CDR}).

We can then use the values in Table 5 as $\{r_{um}\}$ values in (P1)-(P3) to solve them under DEBA decision makers. Let $\mathcal{P}_u$ be the set of all possible pairs of tasks out of $\mathcal{M}_u$ that can be offered to user u. Depending on the user reward allocation class and the rewards offered for each task, the user may be steered to either of the two tasks, or both, or none. This way, the union of possible task pairs in $\mathcal{P}_u$ yields an *inflated* set of tuples $\mathcal{T}_u$. Each tuple t uniquely determines the pair of tasks recommended to user u, the offered rewards for each task, and the induced task choice of the user, $m(t)$. In turn, this determines the quality $q_{um(t)}$ of the contribution and the payment $r_{um(t)}$ to be made to user for this contribution. The app chooses one of these tuples to offer to each user under the budget constraints of each task, in order to maximize the aggregate quality of user contributions to tasks.

Formally, if $x_{ut} = 1,\ t \in \mathcal{T}_u$, when a particular tuple is offered to user u, and $x_{ut} = 0$, otherwise, the optimization problem faced by the app is

$$\max_{\mathbf{x}} \quad \sum_{u \in \mathcal{U}} \sum_{t \in \mathcal{T}_u} q_{um(t)} x_{ut} \tag{13}$$

$$s.t. \quad \sum_{u \in \mathcal{U}} \sum_{t \in \mathcal{T}_u : m(t) = m} r_{um(t)} x_{ut} \leq B_m \quad \forall m \in \mathcal{M} \tag{14}$$

$$\sum_{t \in \mathcal{T}_u} x_{ut} \leq 1 \quad \forall u \in \mathcal{U} \qquad (P4) \tag{15}$$

$$x_{ut} \in \{0, 1\}, \quad u \in \mathcal{U}, t \in \mathcal{T}_u \tag{16}$$

The problem formulation (P4) is a non-trivial variant of the Generalized Assignment problem with multiple-choice type constraints and its solution has independent theoretical interest. However, regarding our problem, we can show that

Theorem 2. *The paired-task offer problem (P4) reduces to the single-task offer problem (P1).*

Proof. Let $\mathbf{x}^*$ be a solution maximizing the objective function of (P1). From that solution we construct an optimal solution for (P4) as follows. For each user u let $m^*(u)$ be the task assigned to u by $\mathbf{x}^*$. We choose any other task m_2 and propose the tuple of tasks $(\mathbf{x}^*(u), m_2)$ to user u, with rewards $r_{um^*(u)}$ as in Table 1, and r_{um_2} less than the value given by (3) if the task is in acceptable distance from the user, else arbitrary. In this way it is obvious that the user will not choose task m_2. We claim that the above solution maximizes the objective function of (P4).

Suppose that the above solution is not optimal, and that the optimal solution to (P4) is achieved by some $\mathbf{x}'$. Assume that this solution proposes to user u the tuple of tasks $t = (m^u_a, m^u_b)$ and that, without loss of generality, the user chooses task m^u_a. We introduce a new vector $\mathbf{x}''$ such that for all $u \in \mathcal{U}$ and $m \in \mathcal{M}$ $\mathbf{x}''_{um} = 1$ if user u chooses task m, else $\mathbf{x}''_{um} = 0$. Since u chooses m^u_a, for sure it was proposed to him with a reward at least as high as what is given in Table 1, *i.e.*, $r_{um(t)} \geq r_{um^u_a}$.

The same holds for all users, thus $\sum_{u \in \mathcal{U}} \sum_{t \in \mathcal{T}_u : m(t) = m} r_{um(t)} x'_{ut} \leq B_m \Rightarrow \sum_{u \in \mathcal{U}} r_{um} x''_{um} \leq B_m \ \ \forall m \in \mathcal{M}$. Thus, $\mathbf{x}''$ is a feasible solution to (P1) achieving higher aggregate quality than $\mathbf{x}^*$, which is a contradiction since $\mathbf{x}^*$ is an optimal solution to (P1). □

In fact, what we proved is even stronger: (P1) and (P4) are computationally equivalent (the first reduces to the second and vice versa).

Corollary 2. *No better result for the paired task offer problem is possible other than solving the single task offer problem and complementing the resulting offers with additional ones that the users will certainly decline.*

Proof. It is apparent from the previous proof, that any better solution to (P4) would result to a better-than-the-optimal solution to (P1), leading to contradiction. □

Notably, the proofs generalize to when more than two tasks are simultaneously proposed to users. A platform then should "pack" tasks together in an offer and choose rewards so that only $m^*(u)$ is acceptable by the user and the rest are not. In all cases, we need Table 5, *i.e.*, the minimum rewards that induce contributions from each user to any task.

6 Discussion and Open Issues

6.1 Validating and training the cognitive heuristics

Throughout the paper, we have *assumed* (a) the relevance of FFTs (and DEBA) in the MCS setting, and (b) that the MCS platform knows the FFT model for each user.

Validating the first assumption would demand statistically large datasets from real MCS applications, which are missing in literature. Beyond the scale and population coverage requirements (number and type of users, duration), such datasets should realize the choice settings (task offers, rewards) that are described in this study. This requirement to run what is essentially a controlled experiment at a large-scale make this task a very challenging one. Alternatives such as online questionnaires and surveys or experimental apps lack in realism and scale and/or suffer from biases in the considered populations (*e.g.*, students) [22], [23].

Given such datasets, the cognitive heuristic models can be *trained* for each user leveraging customized machine learning techniques, which fall under the broader domain of supervised learning [24]. As training sample could serve past user responses to actual task offers or hypothetical ones made to users as part of a calibration process triggered upon their registration with the MCS platform. Parameters to learn out of the model training process include the set of relevant cues and their rank, the acceptability thresholds for continue cues and the type of inspection outcome (positive, negative) that results in a choice at each FFT level.

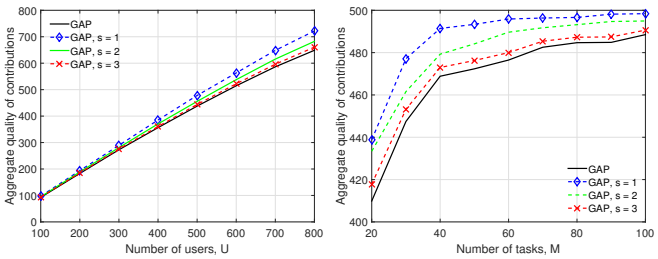
In [11], two heuristic algorithms are proposed for extracting the cue rank and the type of inspection for each cue. Both algorithms use the *medians* of the cue values in the training sample as acceptability threshold values for the cue. In [24] the cue order in the FFT, the inspection type that results in a decision at each level *and* the acceptability thresholds are simultaneously determined out of a more computationally intensive learning process with enumeration flavor.

6.2 Catering for model stochastic effects

6.2.1 Deviations from FFT model prescriptions

FFTs are deterministic models of human decision-making. In practice, however, a user may deviate from her usual behaviour and not follow the model prescriptions. We can model this phenomenon by introducing a probability $a_u \in [0, 1]$ that user u deviates from the assumed FFT model. Setting $a = \max_{u \in \mathcal{U}} a_u$ we can prove the following.

Proposition 2. *The deterministic model achieves at least a* $(1 - a)$ *approximation of the real outcome, assuming that the players follow the proposed FFT model with probability* $(1 - a_u)$.



(a) $M = 25$, variable # users (b) $U = 500$, variable # tasks

Fig. 9: The impact of normally distributed $\{q_{um}\}$ values on the approximately optimal solution of (P1): $R = 1000$, $\theta_r^u \sim \mathcal{U}(0.5, 3.5)$, $\theta_d^u \sim \mathcal{U}(170, 1000)$, $r_{min} = 0.25$.

Proof. The worst thing that can happen under such model deviations is that a user rejects a task offer that she would accept according to the FFT model. Let y_{um} be a random variable that takes the value 1 or 0 depending on whether the player u will undertake the task m, taking into account the solution of (P1) and the probability a_u that she does not follow the model. Namely, $y_{um} = x^*_{um}$ if the user follows the model, else $y_{um} = 0$. Then the expected value of our objective function becomes

$$E[\sum_{u,m} y_{um} q_{um}] = \sum_{u,m} q_{um} E[y_{um}] \geq (1 - a) \sum_{u,m} q_{um} x^*_{um}$$

□

This analysis is meaningful when the parameters a_u, and hence a, are adequately small. A large a_u value would suggest that the FFT is not a good model for the user at hand and would call for research on alternative (deterministic or purely probabilistic) models for capturing her choices. In practice, the parameters a_u could correspond to the training accuracy figures that emerge out of the FFT model training process (see section 6.1).

6.2.2 Variations in the quality of user contributions

Likewise, we can reasonably assume that the quality of users' contributions is not constant and perfectly known to the MCS platform. Instead, the platform *estimates* it taking into account the task specification and information about the user profile and the technical characteristics of her smart device. For any given task m, relevant skills of user u can be inferred from her involvement in related special interest social media, usage stats of mobile apps or explicit inputs about her hobbies.The smart device capabilities, on the other hand, are reflected in its technical specifications, *e.g.*, camera resolution or sensors' precision standards. User skills and device capabilities could then be quantified on a simple rating scale and combined to yield an estimate about q_{um}.

We can model the $\{q_{um}\}$ values as normally distributed random variables $Z_{um} \sim \mathcal{N}(\mu_{um}, \sigma^2_{um})$, a larger mean μ_{um} and smaller standard deviation σ_{um} denoting that user u is better qualified for task m. We can then revisit the problems we formulated in sections 3 and 4.

In Fig. 9, we complement the simulation runs of Fig. 2 with two additional sets of runs. In one of them, we first solve (P1) with $q_{um} \equiv \mu_{um}$, then we draw samples $\{z_{um}\}$ of the variables $\{Z_{um} : x_{um} = 1\}$ for the actually evidenced quality of the user contribution and compute the aggregate

quality accordingly. In the other set, we first sample all normal variables and then solve (P1) with $q_{um} \equiv z_{um}$. This second experiment has theoretical value: it yields an estimate of what additional benefit we could obtain had we known in advance the precise quality of user contributions.

In both cases, we set $\sigma_{um} = (1 - \mu_{um})/s$. Hence, the variance of contributions' quality decreases with the mean, whereas s is a scale parameter that controls horizontally the quality variation around its mean for all users. Two remarks are due out of Fig. 9. First, the curves obtained with the first set of runs practically coincide with those corresponding to deterministic $\{q_{um}\}$. The variations of the random variables around their mean cancel out and the overall sum tends to the quantity $\sum_{u,m} x^*_{um}\mu_{um}$, $\{x^*_{um}\}$ being the matches we determine when solving (P1) with $\{\mu_{um}\}$. Second, the theoretical gain in terms of aggregate quality of task contributions, under perfect knowledge of the instantaneous q_{um} values, is relatively small, even for high variance (s=1), and fades out fast, practically disappearing for $s > 3$.

6.3 From individual user skills to MCS task quality

Building on the analysis in 6.2.2, each q_{um} could be viewed as the expected value of a latent random variable $\mathcal{X}_{um}$ with normal distribution, $X_{um} \sim \mathcal{N}(_{um}, \sigma^2_{um})$. Our modeling has implicitly assumed that the overall task quality obtained out of the individual contributions exhibits *cumulative effects*, *i.e.*, it equals the sum of the expected values of those contributions. This is why the $\{q_{um}\}$ values are added in the objective (1) of (P1) or in the constraint (9) of (P2).

In general, the way the total benefit for the task relates to individual contributions it attracts may vary. Formally, this can be expressed by a set function $\mathcal{F} : \mathcal{P}(\mathcal{U}) \rightarrow R$, which maps the quality of task contributions from any subset S of users, $\mathcal{S} \subseteq \mathcal{U}$, to a total quality for the task.

One extension to the work in this paper would be to revise (P1) and (P2) under different instances of the function $\mathcal{F}$, *e.g.*, when its value for a task m is the maximum quality of all attracted contributions. This would entail answering how the aggregate task quality metric relates to (some function of) $\mathcal{X}_{um}$ (see *e.g.*, [25] and the related thread on the team selection problem).

6.4 Implications for other applications areas

As a final note, bounded rationality bears implications for a broader set of application areas beyond mobile crowdsensing, in which end-users make choices through the mediation of mobile platforms. For example, in smart-energy apps, energy-saving recommendations and consumption plans are issued to the user through the app with the goal to optimize energy savings. In mobile advertising, ads or offers are projected to users, and the aim is to optimize revenue through user response to ads. Common to these platforms is the mission to *engineer* the offered alternatives to users, exploiting recommender systems' practices, and tailor incentives to user preferences so as to nudge them towards desirable choices for the user and platform welfare.

7 Related work

Research on mobile crowdsensing has flourished over the last fifteen years. Recent surveys such as [1] and [2] provide exhaustive records of the different paths the research on MCS optimization has taken, including different objectives (*e.g.*, minimize the number of participants, maximize number of accomplished tasks, minimize budget expenditure); different concerns (*e.g.*, privacy, truthfulness); and the use of different tools (*e.g.*, mechanism design) for the static (offline) or dynamic (online) treatment of the optimization problems. Our work is particularly relevant to studies that explicitly consider the quality of user contributions either as a constraint or as an objective in their work.

Hence, in [26], [27] payment methods are studied as a function of the quality of user contributions. This is in stark contrast with our work, where the minimum required rewards that elicit user contributions are a function of other cues and they are allocated to maximize the aggregate quality of user contributions to tasks. On the other hand, revenue maximization in MCS platforms is ofter pursued through Knapsack-based formulations, *e.g.*, in [28]. Contrary to our formulations for for-profit platforms, these studies introduce constraints on the number of recruited participants rather than on the individual task quality.

In [29] the authors overview existing work on assessing Quality of Information and propose a framework to enforce it. The possibility to infer and predict context and user behavior (like user destinations) is studied in [30], [31]. Contrary to these studies, we use the acquired knowledge about user behaviour to infer minimum prices. Fewer studies have devoted effort to learning user preferences and inferring decision-making processes, either through online questionnaires [32] or combining real data from social networks [3].

One step closer to the work in this paper are the studies in [33], [34], [35], which address the task allocation process for (data) quality maximization under monetary or effort-related budget constraints. The participant selection task leverages GAP-based formulations and corresponding algorithms in [36], whereas it explicitly takes into account distance, energy and sociability in [37], [38]. Contrary to these last few studies, as well as all the aforementioned ones, our starting point is the accumulated experimental evidence about the bounded rationality of human choices. Hence, we model MCS users after heuristics from the field of cognitive psychology and let these models drive the optimization of the MCS platform's operations.

8 Conclusions

Concepts from behavioral science remain, to the best of our understanding, largely unexploited by the (wireless) networking community. In this work, we have made a first attempt to accommodate the extensive experimental evidence on the bounded rationality of human-decision making in the problem of MCS task recommendation and incentive allocation. We have relied on heuristic models from the field of cognitive psychology to this end.

We use those models to infer minimum rewards that can be offered by the platform to elicit users contributions to MCS tasks and reduce the optimization problems faced by the platform to generalized assignment and other (bi)linear problems. This approach leads to significantly better results for both non-profit and for-profit platforms when compared to solutions that do not account for the bounded rationality of the human decision-making.

ACKNOWLEDGMENT

The research work of M. Karaliopoulos and E. Bakali is funded by the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under grant agreement No 892.

REFERENCES

[1] Y. Liu, L. Kong, and G. Chen, "Data-oriented mobile crowdsensing: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2849–2885, 2019.

[2] A. Capponi *et al.*, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2419–2465, 2019.

[3] P. Micholia *et al.*, "Incentivizing social media users for mobile crowdsourcing," *Int. J. Hum.-Comput. Stud.*, vol. 102, no. C, pp. 4–13, Jun. 2017.

[4] T. Luo *et al.*, "Sustainable incentives for mobile crowdsensing: Auctions, lotteries, and trust and reputation systems," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 68–74, Mar. 2017.

[5] H. A. Simon, "Rational choice and the structure of the environment," *Psychol. Rev.*, vol. 63, no. 2, pp. 129–138, 1956.

[6] J. W. Payne, E. Johnson, and J. Bettman, *The Adaptive Decision Maker*. Cambridge, England: Cambridge University Press, 1993.

[7] D. Kahneman, *Thinking, Fast and Slow*. New York: Farrar, Straus and Giroux, 2011.

[8] G. Gigerenzer and D. G. Goldstein, "Reasoning the Fast and Frugal Way: Models of Bounded Rationality," *Psychol. Rev.*, vol. 103, no. 4, pp. 650–669, 1996.

[9] A. Dieckmann, K. Dippold, and H. Dietrich, "Compensatory versus noncompensatory models for predicting consumer preferences," *Judgm. Decis. Mak.*, vol. 4, no. 3, pp. 200–213, April 2009.

[10] L. Martignon, O. Vitouch, M. Takezawa, and M. R. Forster, "Naive and yet enlightened: From natural frequencies to fast and frugal decision trees," *Thinking: Psychological Perspectives on Reasoning, Judgment and Decision Making, Ch. 10*, pp. 189–211, 2005.

[11] L. Martignon, K. V. Katsikopoulos, and J. K. Woike, "Categorization with limited resources: A family of simple heuristics," *J. Math. Psychol.*, vol. 52, no. 6, pp. 352 – 361, 2008.

[12] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York, USA: John Wiley & Sons, Inc., 1990.

[13] N. E. Young, "Sequential and parallel algorithms for mixed packing and covering," in *Proc. 42nd IEEE FOCS*, Las Vegas, NV, USA, Oct. 2001, pp. 538–546.

[14] R. M. Hogarth and N. Karelaia, "Simple models for multiattribute choice with many alternatives: When it does and does not pay to face tradeoffs with binary attributes?" *Manage. Sci.*, vol. 51, no. 12, pp. 1860–1872, 2005.

[15] K. V. Katsikopoulos, "Psychological heuristics for making inferences: Definition, performance, and the emerging theory and practice," *Decis. Anal.*, vol. 8, no. 1, pp. 10–29, Mar. 2011.

[16] M. Karaliopoulos, I. Koutsopoulos, and L. Spiliopoulos, "Optimal user choice engineering in mobile crowdsensing with bounded rational users," in *Proc. IEEE INFOCOM '19*, Paris, France, Apr. 2019, pp. 1054–1062.

[17] L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko, "Tight approximation algorithms for maximum general assignment problems," in *Proc. 17th ACM-SIAM SODA*, Miami, FL, USA, 2006, pp. 611–620.

[18] R. Cohen, L. Katzir, and D. Raz, "An efficient approximation for the generalized assignment problem," *Inf. Process. Lett.*, vol. 100, no. 4, pp. 162 – 166, 2006.

[19] S. G. Kolliopoulos and N. E. Young, "Approximation algorithms for covering/packing integer programs," *J. Comput. Syst. Sci.*, vol. 71, no. 4, pp. 495 – 505, 2005.

[20] G. Gallo and A. Ülkücü, "Bilinear programming: An exact algorithm," *Math. Program.*, vol. 12, no. 1, pp. 173–194, Dec 1977.

[21] G. McCormick, "Computability of global solutions to factorable nonconvex programs: Part i. convex underestimating problems," *Math. Program.*, vol. 10, p. 147175, 1976.

[22] M. Karaliopoulos, L. Spiliopoulos, and I. Koutsopoulos, "Factoring the human decision-making limitations in mobile crowdsensing," in *Proc. IEEE/IFIP WONS 2016*, Italy, January 2016.

[23] G. Cardone, A. Corradi, L. Foschini, and R. Ianniello, "Participact: A large-scale crowdsensing platform," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 01, pp. 21–32, jan 2016.

[24] N. Phillips, H. Neth, J. Woike, and W. Gaissmaier, "Fftrees : A toolbox to create, visualize, and evaluate fast-and-frugal decision trees," *Jdgmt. and Dec. Mak.*, vol. 12, pp. 344–368, 7 2017.

[25] J. Kleinberg and M. Raghu, "Team performance with test scores," in *Proc. 16th ACM EC*, Portland, Oregon, USA, 2015, pp. 511–528.

[26] C.-K. Tham and T. Luo, "Quality of contributed service and market equilibrium for participatory sensing," *IEEE Trans. Mob. Comput.*, vol. 14, no. 4, pp. 829–842, April 2015.

[27] D. Peng, F. Wu, and G. Chen, "Pay as how well you do: A quality based incentive mechanism for crowdsensing," in *Proc. 16th ACM Mobihoc*, Hangzhou, China, June 2015, pp. 177–186.

[28] N. D. Lane *et al.*, "Piggyback crowdsensing (pcs): Energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities," in *Proc. 11th ACM SenSys (Online)*, 2013.

[29] F. Restuccia, N. Ghosh, S. Bhattacharjee, and S. Das, "Quality of information in mobile crowdsensing: Survey and research challenges," *ACM Transactions on Sensor Networks*, vol. 13, 09 2017.

[30] V. Pejovic and M. Musolesi, "Anticipatory mobile computing: A survey of the state of the art and research challenges," *ACM Computing Surveys*, vol. 47, 06 2013.

[31] N. Bui *et al.*, "A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1790–1821, 2017.

[32] M. Karaliopoulos, I. Koutsopoulos, and M. Titsias, "First learn then earn: optimizing mobile crowdsensing campaigns through data-driven user profiling," in *Proc. 17th ACM MobiHoc*, Paderborn, Germany, July 2016, pp. 271–280.

[33] J. Wang *et al.*, "Multi-task allocation in mobile crowd sensing with individual task quality assurance," *IEEE Transactions on Mobile Computing*, vol. 17, no. 9, pp. 2101–2113, 2018.

[34] L. Pu, X. Chen, J. Xu, and X. Fu, "Crowdlet: Optimal worker recruitment for self-organized mobile crowdsourcing," in *Proc. 35th IEEE INFOCOM*, 2016, pp. 1–9.

[35] L. Wang, Z. Yu, D. Zhang, B. Guo, and C. H. Liu, "Heterogeneous multi-task assignment in mobile crowdsensing using spatiotemporal correlation," *IEEE Transactions on Mobile Computing*, vol. 18, no. 1, pp. 84–97, 2019.

[36] M. Tomasoni *et al.*, "Profiling energy efficiency of mobile crowdsensing data collection frameworks for smart city applications," in *6th IEEE MobileCloud*, Bamberg, Germany, March 2018, pp. 1–8.

[37] G. Bajaj and P. Singh, "Load-balanced task allocation for improved system lifetime in mobile crowdsensing," in *Proc. 19th IEEE MDM*, Aalborg, Denmark, June 2018, pp. 227–232.

[38] C. Fiandrino *et al.*, "Sociability-driven framework for data acquisition in mobile crowdsensing over fogcomputing platforms for smart cities," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 4, pp. 345–358, 2017.

Merkouris Karaliopoulos received his Diploma in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Greece, and a Ph.D. in Electronic Engineering from the University of Surrey, UK. Since 2016 he has been a Senior Researcher at the Athens University of Economics and Business, Greece. Prior to that, he held research appointments at the University of Athens (Greece), ETH Zurich (Switzerland), and the University of North Carolina at Chapel Hill (United States). His research interests lie in performance analysis and resource allocation problems emerging in wireless networks and online/mobile platforms.

Eleni Bakali received her Diploma in Applied Mathematical and Physical Sciences, and her Ph.D. in Computational Complexity from the National Technical University of Athens (NTUA), Greece. Since 2019 she is a postdoctoral researcher at the Athens University of Economics and Business, and since 2020 she is a postdoctoral researcher at the NTUA. Her research interests lie in algorithms, complexity, and randomness in computation.