Coverage-aware Stable Task Assignment in Opportunistic Mobile Crowdsensing

Fatih Yucel, *Member, IEEE,* Murat Yuksel, *Senior Member, IEEE,* and Eyuphan Bulut, *Senior Member, IEEE,*

***Abstract*—In opportunistic mobile crowdsensing, the objective of service requesters is to have as many of their sensing tasks completed as possible within their budget constraints, whereas that of participants (workers) is to collect the highest monetary reward possible on their trajectories. However, these objectives can conflict and may result in unhappy service requesters or workers if the matching between them is not handled carefully. In this paper, we study the problem of finding task assignments that fulfill both coverage-aware preferences of service requesters and profit-based preferences of workers in a budget-constrained, opportunistic mobile crowdsensing system. Since this is a matching problem with bilateral preferences, we aim to find a matching in which everyone is satisfied with their assignment based on their preference profile. We first propose a polynomial-time approximation algorithm for general settings, and then show that a slightly modified version of this algorithm has a constant approximation ratio when the rewards offered to workers by service requesters are proportional to the coverage capability of workers for corresponding tasks. Through extensive simulations, we evaluate the performance of our algorithms in different settings, and show that they mostly provide substantially better task assignments in terms of user happiness and coverage quality while having a few orders of magnitude lower running times compared to the benchmark algorithms.**

***Index Terms*—Mobile crowdsensing, task assignment, budgeted coverage maximization, stable matching.**

I. Introduction

Mobile Crowdsensing (MCS) is a social business model [1] that enables people to complete their sensing tasks in an efficient manner by recruiting mobile, interested users. It does not only serve the needs of task requesters, but also provides interested users with alternative means to make a profit by carrying out the requested sensing tasks, which generally take a very small amount of time and do not require any expertise, such as taking pictures of buildings for place naming systems [2]. The three main entities in a typical MCS system are a crowdsensing center/platform, task requesters, and workers (i.e., users that are interested in performing tasks voluntarily or in exchange for a reward). The key roles of the crowdsensing center are to make the information of the requested sensing tasks and active workers public (i.e., their spatiotemporal and financial constraints), and to assist in finding an efficient task assignment by either directly producing a matching between the tasks and workers based on the given constraints or by providing a platform for the task requesters and workers to decide on a matching in a distributed fashion.

The tasks in an MCS system can be performed in a *participatory* [3] or *opportunistic* [4] manner (i.e., explicit/implicit user participation, respectively). In the former, workers are required to interrupt their daily schedules and carry out the assigned tasks at the specified time by traveling to the task regions. Hence, this approach comes with a travel cost, which tends to dominate the other costs to be incurred by a worker such as communication/storage cost and energy consumption on the sensing device. On the other hand, in the latter, workers maintain their schedules and perform the assigned tasks opportunistically when they happen to be in the task regions. Therefore, in this type of sensing, the task assignments should be made in a way that ensures the workers that are assigned to a task are likely to visit the task region within an acceptable time frame. Besides, although there is no travel cost associated with the tasks in the opportunistic sensing, it usually takes longer for a worker to complete the assigned tasks compared to the participatory sensing.

Regardless of the adopted sensing strategy (participatory or opportunistic), the key factor that defines the performance of the overall MCS campaign is the efficiency of the task assignments. Yet, the definition of efficiency (i.e., optimization criteria) varies widely across different studies. Some of the objectives considered in the MCS literature are minimizing the travel costs of workers [3], [5], completing the tasks in the shortest time possible [6], maximizing the quality of service received by task requesters [7], minimizing the energy consumption [8], and maximizing the number of completed tasks [9].

A particularly important objective in the opportunistic MCS systems is to maximize the sensing coverage over a set of points of interest (POIs), which has recently been studied in [10]–[13]. However, these studies either do not consider budget constraints of task requesters or assume that there is only a single task requester (i.e., a single budget constraint) in the system. This may not be a practical assumption as there can be multiple task requesters with a unique set of goals and an individual budget constraint. Moreover, some task requesters may prefer to allocate a separate budget for different sets of POIs.

Another issue with the existing task assignment frameworks is that they disregard individual user preferences in the assignment process in order to optimize the task assignment based on the aforementioned system-level utility metrics. Consequently, they are likely to produce dissatisfying assignments for the users, which may be detrimental for the long-term operational

F. Yucel and E. Bulut are both with the Department of Computer Science, Virginia Commonwealth University, Richmond, VA, 23284. E-mail: {yucelf, ebulut}@vcu.edu.

M. Yuksel is with the Department of Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816. E-mail: {Murat.Yuksel}@ucf.edu.

well-being of the system as the users that repeatedly get such assignments may stop using the system altogether. For instance, it has been shown in [14] that a task assignment algorithm that maximizes the number of assigned tasks ends up making up to 35% of all worker-task requester pairs unhappy with their assignments.

Given a two-sided matching, an unhappy pair is defined as a pair of individuals who are not assigned to each other, but would prefer each other to their current assignments. A matching with no unhappy pairs is said to be stable [15], because absence of unhappy pairs indicates that there is no pair who would reject their assignments to get matched with each other. Besides this equilibrium state, considering user preferences in a matching problem also has the advantage of enabling each individual to specify her own interests and constraints in the matching process via a preference profile/list (e.g., preference for tasks that are close to their home or work).

There are recent studies [14], [16]–[18] that integrate the concept of stability in the task assignment process. However, these studies fail to consider some crucial aspects of task assignments. For example, [14] assumes simple tasks that can always be completed by a single worker, and [18] does not consider budget constraints. Moreover, they all assume participatory MCS systems and only consider additive utility functions (i.e., the total utility of a set of workers for a task is equal to the sum of their individual utilities). Yet the coverage over a set of POIs is inherently non-additive because of the potential overlaps among the POIs covered by different workers.

We can summarize the key issues that need to be taken into consideration in task assignments in opportunistic MCS and the studies that partly addressed them as follows:

- *Task requester preferences [10]–[13]:* Each task requester desires to have a matching that maximizes the coverage over the POIs that her task needs.
- *Budget feasibility [11]–[13], [16], [17]:* Each task requester has a budget constraint which should not be violated.
- *Worker preferences [14], [16]–[18]:* Each worker desires to maximize his net profit from the system in each task assignment period.
- *Stability [14], [16]–[18]:* Since the objectives above are likely to be in conflict with each other, they should be achieved in a fair way that results in as few unhappy users as possible.

In this paper, we address all of these issues together and make the following contributions:

- We formally define the stability conditions for task assignments in coverage-aware, opportunistic MCS systems with budget constraints.
- We prove that a fully stable task assignment may not exist in some MCS instances, and it is NP-hard even to check whether one exists in a given instance.
- We present a polynomial-time approximation algorithm for the stable task assignment problem, and prove that it always produces $\frac{4}{1-\rho}-$stable matchings, where ρ is the largest reward to budget ratio (normalized between 0 and 1) in the system.
- We show that a variant of our algorithm has an approximation ratio of 5 in MCS systems with proportional reward schemes, where the rewards offered to the workers are proportional to the utility they provide for the tasks.
- We compare the performance of our algorithms with two benchmark algorithms proposed in [11] and [12] via real-data based, extensive simulations, and show that our algorithms produce significantly better task assignments in terms of both user happiness (up to 25%) and achieved coverage (up to 18%), and run up to four orders of magnitude faster compared to the benchmark algorithms in most settings.

The rest of the paper is structured as follows. In Section II, we present a summary of the related work. In Section III, we introduce the system model. In Section IV, we first present a formal problem definition and discuss the hardness of the problem. We also show that a stable matching may not exist in some MCS instances. Then, we describe our approximation algorithms and derive their approximation ratios. In Section V, we evaluate the performance of our algorithms through simulations. Finally, in Section VI, we provide our conclusions.

II. Related Work

In MCS systems, the two key problems are to incentivize the users to take part in the MCS campaign and to assign registered users in the system to the available tasks. In the MCS literature, these problems have been addressed by *incentive mechanisms* [19], [20] and *task assignment or allocation algorithms* [21], [22], respectively.

Incentive mechanisms: In [23], an auction-based incentive mechanism is proposed, where the users whose submissions for the available tasks are accepted by the corresponding task requesters are paid a reward based on the quality of their submission (i.e., the sensed data). [24] and [25] also propose quality-aware incentive mechanisms. Specifically, [24] focuses on the scenario where workers arrive in an online manner, while [25] considers the temporal changes in the quality of workers for the crowdsourcing campaign in each run of the task allocation (based on their performance in previous runs). [26] presents incentive mechanisms for cooperative mobile crowdsensing in online social networks, where tasks require multiple workers for successful completion, and each worker is able to specify a set of workers that he wants to cooperate with on performing tasks (e.g., friends in social networks). [27] proposes an incentive mechanism that exploits the concepts of reference effect and loss aversion from behavioral economics, and addresses the problem of completing tasks in regions that are not visited frequently by participants.

Task assignment: [28]–[31] investigate the task assignment (or worker recruitment) problem. [28] transforms the assignment problem into a scheduling problem based on the expiration time of tasks, and aims to minimize the total delay in completion of tasks. The goal in [29] is to maximize the number of completed tasks while keeping the total travel distance of workers as low as possible. [30] and [31] study

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2021.3065688, IEEE Transactions on Vehicular Technology

the problem in an online setting. [30] utilizes the branch-and-bound method [32] to find an assignment that maximizes the total profit of workers, while [31] proposes four different online algorithms to maximize the sensing quality received by task requesters. There are some recent studies that explore the privacy and reliability issues in task assignments, as well. For instance, [33] and [34] both address the privacy-preserving task assignment problem. Particularly, [33] focuses on the worker privacy and assumes a system model, in which the quality scores of workers are not disclosed to the platform, while [34] considers the privacy of task requesters as well as that of workers. On the other hand, [35] investigates the reliability issue in task assignments, and proposes a personalized recommendation system that takes users' reliability for each task into account in estimating their suitability for the tasks. Our work differs from these prior efforts by considering coverage-awareness as part of the design.

Coverage-awareness: There are also a number of studies that address the issue of coverage-aware sensing. For instance, [4] aims to find a minimum-cost user set that will cover all PoIs requested by campaign organizer (who does not have a budget constraint), and assumes that both the sensing and delivery of the sensed data are realized in an opportunistic manner. [10] adopts the same objective, but assumes a participatory MCS system and proposes a reverse auction framework instead of a worker selection algorithm. However, this study does not consider a budget constraint either.

[11], [12] and [13] study the problem of finding a worker set with maximum weighted coverage over a set of PoIs in a budget-constrained MCS system. In particular, [11] considers an opportunistic MCS system with static reward profile for workers, and proposes a polynomial-time approximation algorithm for worker selection. [12] assumes the same system model and presents a greedy algorithm that is shown to perform better than the algorithm in [11] in synthetically generated instances. However, no theoretical performance guarantee has been shown for this algorithm (unlike [11]). On the other hand, [13] proposes a strategy-proof incentive mechanism, in which the reward profile of the recruited workers is dynamically determined based on their bids. A different type of coverage problem is studied in [36], where the goal is to find a small subset of a huge number of photos uploaded by workers, which fully covers a set of targets (i.e., each aspect of a target appears in at least one selected photo).

Preference-awareness: An important issue in the studies mentioned thus far is that they aim to optimize task assignments solely based on a system-level utility metric (e.g., travel cost, QoS, coverage), and neglect to take user (i.e., workers and task requesters) preferences into consideration in the task assignment process. An introductory work to the matching problems with user preferences is [15], which proposes a polynomial-time algorithm, namely *the deferred-acceptance mechanism*, that finds a stable matching (i.e., a matching with no unhappy pairs) between two groups of objects with bilateral preferences. This algorithm is designed for one-to-one matching and many-to-one matching with capacity or quota constraints, and unfortunately cannot be used in the presence of budget constraints. A considerable number of studies have adapted this algorithm to find stable matchings for real-world matching problems in different contexts such as driver-passenger matching in taxi dispatching systems [37], [38] and supply-demand matching for charging of electric vehicles [39], [40].

TABLE I: A summary of the related work (* denotes this paper).

	Coverage-aware	Budget-constrained	Preference-aware
*	✓	✓	✓
[4]	✓	✗	✗
[10]	✓	✗	✗
[11]	✓	✓	✗
[12]	✓	✓	✗
[13]	✓	✓	✗
[14]	✗	✓	✓
[16]	✗	✓	✓
[17]	✗	✓	✓
[18]	✗	✗	✓
[23]	✗	✓	✗
[24]	✗	✓	✗
[25]	✗	✓	✗
[26]	✗	✗	✗
[27]	✓	✗	✗
[28]	✗	✗	✗
[29]	✗	✗	✗
[30]	✗	✗	✗
[31]	✗	✗	✗
[33]	✗	✓	✗
[34]	✗	✓	✗
[35]	✗	✗	✓

Moreover, a few studies consider user preferences and stable matchings in task assignments in MCS. [14] studies the problem of finding a maximum size one-to-one matching between workers and tasks with minimum instability (i.e., with as few unhappy pairs as possible). The problem turns out to be NP-hard, thus the authors propose two different polynomial-time heuristic algorithms. [16]–[18] focus on the many-to-one stable task assignments with additive utility functions. [17] considers a budget-constrained MCS system where the quality of a worker is identical for all tasks, and presents an exponential-time algorithm to find weakly-stable matchings. [16] provides the nonexistence and hardness results for the generalized version of the problem studied in [17], and presents pseudo-polynomial time approximation and heuristic algorithms for different settings. Differently, [18] assumes that task requesters have a quota constraint, and uses a variant of the deferred-acceptance mechanism to find stable task assignments. Note that none of these studies addresses the issue of coverage-aware sensing. Similar to [17], some studies in the stable matching literature [41], [42] investigate the many-to-one, budget-constrained stable matching problem with additive utility functions. However, to the best of our knowledge, there is no study that considers coverage-based or non-additive utility functions in that literature, either. Hence, this study is the first to address the problem of both coverage and user preference-aware task assignment in MCS. A summary of the related work with respect to the three most relevant design parameters is presented in Table I.

III. System Model

We assume a system model with a matching platform that receives sensing task requests $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ over a set of PoIs $\mathcal{P} = \{p_1, p_2, \ldots, p_k\}$, and determines the assignments between these tasks and workers (data contributors). Each task t needs a type of sensed data from a certain subset of PoIs, denoted by $P(t) \subseteq \mathcal{P}$. For a task, some of the PoIs might be more important than the others due to their spatial features (e.g., being close to a production plant for an air pollution sensing task), thus we also let each task t assign a weight $v_t(p)$ to each PoI p in $P(t)$.

Let $\mathcal{W} = \{w_1, w_2, \ldots, w_m\}$ be the set of registered workers in the system. We assume that workers are not willing to interrupt their daily schedule, but they accept to perform the tasks on their trajectories, i.e., *opportunistic sensing*. According to the frequency of task assignments and the nature and time sensitivity of tasks in the system, a different portion and timescale of their future trajectories (e.g., daily, hourly) can be considered in the task assignment process. Let X_w be the set of all locations that will be visited by worker w during the considered time frame. Similar to the previous work [11], we assume that a PoI is covered by worker w if it falls in the sensing range d_w of the worker. Then, the set of PoIs that are covered by worker w is given by

$$C(w) = \{p \in \mathcal{P} : d(p, x) \leq d_w, \exists x \in X_w\}, \tag{1}$$

where $d(p, x)$ is the Euclidean distance between the PoI p and $x \in X_w$. The coverage set of worker w for task t can then be defined as:

$$C_t^w = C(w) \cap P(t) \tag{2}$$

If the requester of task t needs to have some of the POIs sensed by a deadline that is earlier than the end of the current assignment cycle, and worker w will not arrive at the corresponding locations in time according to his trajectory, then we can simply remove these POIs from C_t^w. The total utility of a set S of workers for task t is equal to their total weighted coverage over $P(t)$, which can be calculated by

$$U_t(S) = \sum_{p \in C_t^S} v_t(p), \text{ where } C_t^S = \cup_{w \in S} C_t^w \tag{3}$$

Consider the instance illustrated in Fig. 1, and let t be a task in this instance with the following properties:

$$P(t) = \{p_1, p_2, p_3, p_4\},$$
$$v_t(p_i) = v_t(p_j) = 1, \forall i, j \in \{1, 2, 3, 4\}.$$

Then, the individual utilities of workers w_1 and w_2 for task t would be $U_t(w_1) = 3$ and $U_t(w_2) = 2$, since they cover 3 and 2 of the PoIs requested in task t, respectively. Their joint utility for task t would be $U_t(\{w_1, w_2\}) = 4$, which is evidently less than the sum of their individual utilities. This demonstrates the non-additiveness of the utility function given in Eq. 3.

Moreover, we assume a budget-constrained system model with monetary incentives, where the requester of task t has a budget b_t that limits the amount of monetary incentives to be spent for the completion of task t, and offers each eligible worker w a reward $r_t(w)$ ($\leq b_t$) to cover the PoIs in C_t^w. Let $r_t(S)$ be the total rewards offered to the worker set S (i.e., $r_t(S) = \sum_{w \in S} r_t(w)$). Besides, for each worker w, there is a cost $c_t(w)$ associated with each task t, which worker w can estimate considering the factors such as cost of delivering the sensed data to the task requester via cellular networks, energy consumption due to sensing, and privacy risks.

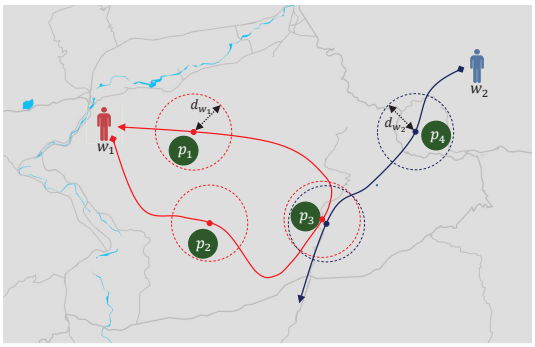


Fig. 1: An MCS instance with 2 workers (w_1, w_2) and 4 PoIs (p_1, p_2, p_3, p_4). The trajectories of workers are shown with solid lines.

Let $\mathcal{M}$ be a matching between the tasks and the workers in the system. Also, let $\mathcal{M}(u)$ denote the assigned task (worker set) to worker (task) u in $\mathcal{M}$. In order for $\mathcal{M}$ to be a feasible and individually rational matching, it should satisfy the following conditions:

- a worker w is either unmatched or matched with a task, i.e.,

$$\mathcal{M}(w) = \emptyset \text{ or } \mathcal{M}(w) \in \mathcal{T}, \tag{4}$$

- a task is matched with a subset of workers (which may be $\emptyset$), i.e.,

$$\mathcal{M}(t) \subseteq \mathcal{W}, \tag{5}$$

- if worker w is matched with task t, the worker set of task t also includes worker w, and vice versa, i.e.,

$$\mathcal{M}(w) = t \text{ iff } w \in \mathcal{M}(t), \tag{6}$$

- no worker w is matched with a task that is not economically beneficial for him, i.e.,

$$r_t(w) > c_t(w) \text{ if } \mathcal{M}(w) = t, \tag{7}$$

- and, no task t is matched with a set of workers that she cannot afford, i.e.,

$$\sum_{w \in \mathcal{M}(t)} r_t(w) \leq b_t. \tag{8}$$

The worker-task pair (w, t) is said to be a *qualified* pair if there exists a feasible and individually rational matching, in which worker w is matched with task t. A summary of the key notations is presented in Table II.

IV. Coverage-aware Stable Task Assignment

In this section, we first formally describe the task assignment problem studied in this paper, and then provide the details of the proposed approximation algorithms.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2021.3065688, IEEE Transactions on Vehicular Technology

TABLE II: Key notations.

Notation	Description
$\mathcal{W}$, $\mathcal{T}$, $\mathcal{P}$	Set of workers, tasks and PoIs, respectively
m, n, k	Number of workers, tasks and PoIs, respectively
$\mathcal{M}$	A feasible & individually rational task assignment
$\mathcal{M}(u)$	Assigned task (worker set) to worker (task) u in $\mathcal{M}$
$c_t(w)$	Cost of performing task t for worker w
$r_t(w)$	Reward offered to worker w to perform task t
$g_t(w)$	Profit of worker w from task t $(r_t(w) - c_t(w))$
$r_t(S)$	$\sum_{w \in S} r_t(w)$
$P(t)$	PoI set of task t
$v_t(p)$	Weight of PoI $p \in P(t)$ for task t
$C(w)$	Set of PoIs covered by worker w
C_t^w	$C(w) \cap P(t)$
$U_t(S)$	Utility of $S \subseteq \mathcal{W}$ for task t
b_t	Budget of task t
$b_t^{\mathcal{M}}$	Remaining budget of task t in $\mathcal{M}$
δ_t	Dissatisfaction ratio of task t
L_w	Preference list of worker w
d_w	Sensing range of worker w
ρ	$\max_{w \in \mathcal{W}, t \in \mathcal{T}} r_t(w)/b_t$

A. Problem Statement

Stability is an important concept in matching problems with selfish and rational individuals. It defines the satisfaction of users with their assignments [14] and promotes long-term user participation by making certain that users are not upset by being forced into less favorable assignments whilst there are better options available. Thus, considering stability in the matching process is not only beneficial for task requesters and workers, but also for the platform. Below, we formally define the stability conditions for our settings.

Definition 1 (Unhappy coalition)**.** *Given a matching $\mathcal{M}$, a task t and a subset S of workers form an unhappy coalition (denoted by $\langle S, t \rangle$) if the following conditions hold for a subset S' of the workers assigned to task t in $\mathcal{M}$:*

- *task t would be better off with S than with S', i.e.,*

$$U_t(S \cup (\mathcal{M}(t) \setminus S')) > U_t(\mathcal{M}(t)), \tag{9}$$

- *task t can replace S' with S without violating her budget constraint, i.e.,*

$$r_t(S) - r_t(S') \leq b_t^{\mathcal{M}}, \tag{10}$$

  where $b_t^{\mathcal{M}}$ is the remaining budget of task t in $\mathcal{M}$ (i.e., $b_t^{\mathcal{M}} = b_t - \sum_{w \in \mathcal{M}(t)} r_t(w)$),
- *every worker w in S prefers task t to task t' to whom he is currently assigned in $\mathcal{M}$, i.e.,*

$$\forall w \in S,\ g_t(w) > g_{t'}(w), \tag{11}$$

  where $g_t(w) = r_t(w) - c_t(w)$ is the net profit of performing task t for worker w, and $g_{t'}(w) = 0$ if worker w is currently unmatched (i.e., $\mathcal{M}(w) = t' = \emptyset$).

Given a worker-task pair (w, t), if there exists an unhappy coalition $\langle S, t \rangle$ such that $w \in S$, we call this pair a *coalitionally unhappy pair*.

Definition 2 (Stable matching)**.** *A matching $\mathcal{M}$ is (coalitionally) stable if it does not contain any unhappy coalitions.*

Note that this is the strongest stability definition in many-to-one matching problems (see [41] for weaker stability definitions). Therefore, if a matching is stable, no one in the matching has even a small incentive to deviate from their current assignment. However, even with additive utilities where the total utility of a set of workers for a task is simply the sum of their individual utilities, a stable matching may not exist, and checking the existence (and finding one if exists) is NP-hard [41]. Since non-additive utilities are a more generalized form of additive utilities (i.e., a problem instance with additive utilities can easily be converted to one with non-additive utilities, but not vice versa), we conclude that the same existence and hardness results also apply to the problem of finding stable matchings in our settings where the utilities of workers for tasks are non-additive as we have

$$U_t(\{w_i, w_j\}) < U_t(\{w_i\}) + U_t(\{w_j\}). \tag{12}$$

when $C_t(w_i) \cap C_t(w_j) \neq \emptyset$. The following theorem formally proves nonexistence of optimal solutions in some cases.

TABLE III: An MCS instance where no fully optimal or stable task assignment exists. The weights of the PoIs are identical for both tasks, and $c_w(t) = 0$ for all (w, t) pairs.

		Rewards	
$\mathcal{W}$	$C(w)$	t_1	t_2
w_1	p_1, p_2, p_3, p_4	4	0
w_2	$p_5, p_6, p_7, p_{10}, p_{11}$	3	2
w_3	p_8, p_9, p_{12}	2	3

$\mathcal{T}$	$P(t)$	b_t
t_1	p_{1-9}	5
t_2	p_{10-12}	3

TABLE IV: All feasible matchings that can be defined on the instance given in Table III along with one of the unhappy coalitions they contain and the dissatisfaction ratios of tasks.

$\mathcal{M}$	Unhappy coalition	δ_{t_1}	δ_{t_2}
$t_1 \to \emptyset$; $t_2 \to \emptyset$	$\langle \{w_1\}, t_1 \rangle$	∞	∞
$t_1 \to \emptyset$; $t_2 \to w_2$	$\langle \{w_1\}, t_1 \rangle$	∞	1
$t_1 \to \emptyset$; $t_2 \to w_3$	$\langle \{w_1\}, t_1 \rangle$	∞	2
$t_1 \to w_1$; $t_2 \to \emptyset$	$\langle \{w_2\}, t_2 \rangle$	5/4	∞
$t_1 \to w_2$; $t_2 \to \emptyset$	$\langle \{w_3\}, t_2 \rangle$	5/3	∞
$t_1 \to w_3$; $t_2 \to \emptyset$	$\langle \{w_2\}, t_2 \rangle$	5/2	∞
$t_1 \to w_2, w_3$; $t_2 \to \emptyset$	$\langle \{w_3\}, t_2 \rangle$	1	∞
$t_1 \to w_1$, $t_2 \to w_2$	$\langle \{w_2, w_3\}, t_1 \rangle$	5/4	1
$t_1 \to w_1$, $t_2 \to w_3$	$\langle \{w_2\}, t_2 \rangle$	1	2
$t_1 \to w_2$, $t_2 \to w_3$	$\langle \{w_1\}, t_1 \rangle$	4/3	1
$t_1 \to w_3$, $t_2 \to w_2$	$\langle \{w_2\}, t_1 \rangle$	5/2	1

Theorem 1. *There exist MCS instances that do not allow for a stable matching.*

Proof. We prove by giving an example, which is described in Table III. All of the feasible and individually rational matchings that can be defined on this instance is also provided in Table IV. Since each matching contains at least one unhappy coalition, we conclude that no stable matching exists in this instance. □

Due to the nonexistence and hardness results for stable matchings, we consider the following relaxation. First, let $\mathcal{S}_t$ be the set of all worker sets that form an unhappy coalition with task t in a given matching $\mathcal{M}$. Formally,

$$\mathcal{S}_t = \{G \subseteq \mathcal{W} : \langle G, t \rangle \text{ is an unhappy coalition}\}. \tag{13}$$

Also, $\forall S \in \mathcal{S}_t$, let

$$\mathcal{E}_S = \{E \subseteq \mathcal{M}(t) : r_t(S) \leq b_t^{\mathcal{M}} + r_t(E)\}, \tag{14}$$

and

$$S^R = \arg\min_{E \in \mathcal{E}_S} U_t(E \setminus (S \cup \mathcal{M}(t))). \tag{15}$$

That is, $S^R \subseteq \mathcal{M}(t)$ is the minimum loss worker set that can be replaced by S within the budget constraint of task t. Then, we can calculate the dissatisfaction ratio of task t in this matching by:

$$\delta_t = \begin{cases} 1, & \text{if } \mathcal{S}_t = \emptyset \\ \infty, & \text{if } \mathcal{S}_t \neq \emptyset,\ \mathcal{M}(t) = \emptyset \\ \max\limits_{S \in \mathcal{S}_t} \left\{ \frac{U_t(S \cup (\mathcal{M}(t) \setminus S^R))}{U_t(\mathcal{M}(t))} \right\}, & \text{otherwise.} \end{cases} \tag{16}$$

Note that the minimum value that δ_t can have is 1, which indicates that task t is perfectly happy in the matching. Finally, we can formally define our objective function as:

$$\text{maximize } \min_{t \in \mathcal{T}} \frac{1}{\delta_t}. \tag{17}$$

Consider the instance given in Table III. Based on the dissatisfaction ratios of tasks in different feasible matchings provided in Table IV, the optimal matching with respect to (17) is $t_1 \rightarrow w_1$, $t_2 \rightarrow w_2$ where the value of (17) is 0.8. The following definition will be used hereafter to signify how optimal a matching is in terms of stability.

Definition 3 (α-stable matching)**.** *A matching $\mathcal{M}$ is α-stable ($\alpha \geq 1$) if*

$$\max_{t \in \mathcal{T}} \delta_t \leq \alpha. \tag{18}$$

Reward schemes. We consider two different reward schemes: *general* and *proportional*. In the general scheme, there is not any assumed relation between the rewards a task requester offers to workers and the utility they provide for the relevant task. However, based on the common practice seen in most of the real-world applications (e.g., Amazon MTurk [43]), it is natural to see a correlation between the two. Hence, in the proportional scheme, we assume that for each task t, the amount of rewards offered to the workers are proportional to their utility. That is

$$r_t(w) = \theta_t \times U_t(w), \tag{19}$$

where θ_t denotes the reward per utility value for task t. It should be noted that a different task t' may have a different reward per utility value (i.e., $\theta_t \neq \theta_{t'}$).

B. Approximation Algorithm

The outline of our polynomial-time approximation algorithm is presented in Algorithm 2. The main idea behind it is to check the potential assignments between the qualified pairs following the order in the preference lists of the workers, and make matching decisions by converting the worker selection problem to an online optimization problem. It begins by calling Algorithm 1, which forms the preference list L_w of each worker w[1] (i.e., tasks in non-increasing order of profits they will provide to worker w), and initializes the matching and the other required variables. Throughout its execution, our algorithm maintains a stack that consists of the workers that are unmatched and whose preference lists have not been entirely traversed by the algorithm (i.e., $index_w \leq |L_w|$). In each iteration of the while loop, it pops one (w) of these workers from the stack and attempts to assign him to the

[1]A worker can also form his preference list himself and submit only this list to the platform, if he does not like to disclose his cost (or profit) for each task.

Algorithm 1: Initialize ($\mathcal{W}$, $\mathcal{T}$), $\mathcal{M}$

Input: $\mathcal{W}$: Set of workers
$\mathcal{T}$: Set of tasks
$\mathcal{M}$: Matching between $\mathcal{W}$ and $\mathcal{T}$

foreach $w \in \mathcal{W}$ **do**
  $L_w \leftarrow$ order $\mathcal{T}$ by non-increasing value of $g_t(w)$
  $L_w \leftarrow L_w \setminus \{t \in L_w \mid g_t(w) \leq 0\}$
  $index_w = 1$
  $\mathcal{M}(w) = \emptyset$
end
foreach $t \in \mathcal{T}$ **do**
  foreach $w \in \mathcal{W}$ **do**
    $x_t(w) = 0$, $\eta_t(w) = 0$
    foreach $p \in P(t)$ **do**
      $z_t(p, w) = 0$
    end
  end
  $H_t = \emptyset$
  $\mathcal{M}(t) = \emptyset$
  $\mathcal{A}_t =$ *false*
end

Algorithm 2: StableTaskAssignment ($\mathcal{W}$, $\mathcal{T}$, σ)

Input: $\mathcal{W}$: Set of workers
$\mathcal{T}$: Set of tasks
σ: Reward scheme (general or proportional)

Let $\mathcal{M}$ be a matching between $\mathcal{W}$ and $\mathcal{R}$
Initialize($\mathcal{W}, \mathcal{T}, \mathcal{M}$)
Stack.push($\mathcal{W}$)
while *Stack is not empty* **do**
  $w \leftarrow$ *Stack.pop()*
  if $index_w \leq |L_w|$ **then**
    $t \leftarrow (index_w)$th task in L_w
    $index_w = index_w + 1$
    $\mathcal{M}(w) = t$, $\mathcal{M}(t) = \mathcal{M}(t) \cup \{w\}$
    if σ *is general* **then**
      $\mathcal{R} \leftarrow$*WorkerSelection*$(t, w, \mathcal{M})$
    else
      $\mathcal{R} \leftarrow$*WorkerSelectionProportional*$(t, w, \mathcal{M})$
    end
    foreach $w' \in \mathcal{R}$ **do**
      $\mathcal{M}(t) \leftarrow \mathcal{M}(t) \setminus w'$, $\mathcal{M}(w') = \emptyset$
      Stack.push(w')
    end
  end
end
return $\mathcal{M}$

next task (t) in his preference list. Although the matching is temporarily updated by assigning worker w to task t (line 9), the actual decision of acceptance is made by calling Algorithm 3 or Algorithm 4 (which are described later in this section) according to the adopted reward scheme. These algorithms return the workers ($\mathcal{R}$) that are removed from the current assignment set of task t. Then, the algorithm sets these workers free again and pushes them back onto the stack (lines 15-18).

Below, we present the performance guarantees of the algorithm with both general and proportional reward schemes by leveraging the analogy between the worker selection step of the algorithm (lines 11 and 13) and the online budgeted maximum coverage *(OBMC)* problem. To this end, we first give a brief description of this problem.

OBMC problem: Assume that a predefined budget B and a universal set $\mathcal{U} = \{u_1, u_2, \ldots, u_{\hat{p}}\}$ with associated weights $\{\hat{v}_i : i = 1, \ldots, \hat{p}\}$ are given. In each iteration i, a set $S_i \subseteq \mathcal{U}$ with a cost of c_i is introduced in an online manner, and the objective is to maximize the weighted coverage over $\mathcal{U}$ within the budget constraint by keeping a certain subset of the introduced sets $\mathcal{S} = \{S_1, S_2, \ldots, S_{\hat{n}}\}$. However, the budget limit cannot be exceeded at any time (i.e., the total cost of the retained sets should always be less than B), and a set that has been rejected/preempted at some point cannot be included in the solution later.

Theorem 2. *(Rawitz and Rosen [44]) There is a* $\frac{4}{1-r}$*-competitive online deterministic algorithm for the OBMC problem, where* $r = \max_{S_i \in \mathcal{S}} \frac{c_i}{B}$.

1) General Reward Scheme: We first describe the task selection mechanism for the general reward scheme by giving a pseudo-code description in Algorithm 3. This algorithm is adapted from the OBMC algorithm mentioned in Theorem 2 to our setting (also optimized for running time, which was not a primary concern in [44])). It accepts a new set if the ratio of the additional utility (i.e., weighted coverage) the set will provide to its cost is larger than 2 times the ratio of the total utility to the total cost in the current solution (line 7). If accepting the set violates the budget constraint, the sets in the current solution are discarded one by one in non-decreasing order of efficiency (utility provided per cost) until the budget constraint is satisfied. Another unique aspect of this algorithm is that when it calculates the total utility in the current solution, it also accounts for the distinct utility that was provided by the set that was discarded the latest as if it had been partially kept in the solution. For each task t, the workers in this imaginary solution are stored in H_t, the fraction of C_t^w used in the imaginary solution is stored in $x_t(w)$, the efficiency of a worker w in H_t is stored in $\eta_t(w)$, and the fraction of a PoI p covered by C_t^w is stored in $z_t(p, w)$. Interested readers are referred to [44] for more detailed descriptions of these variables.

Theorem 3. *Algorithm 2 always produces* $\frac{4}{1-\rho}$*-stable matchings, where* $\rho = \max_{w \in \mathcal{W}, t \in \mathcal{T}} (\frac{r_t(w)}{b_t})$*, in an MCS system with a general reward scheme.*

Proof. We prove this by contradiction. Assume that there is an unhappy coalition $\langle \hat{S}, \hat{t} \rangle$ that prevents the final matching produced by the algorithm from being a $\frac{4}{1-\rho}$-stable matching. Thus, based on Definition 3, there must be a set $S' \subseteq \mathcal{M}(\hat{t})$ such that

$$U_{\hat{t}}(\hat{S} \cup (\mathcal{M}(\hat{t}) \setminus S')) > \frac{4 \times U_{\hat{t}}(\mathcal{M}(\hat{t}))}{1-\rho} \quad \text{and } r_{\hat{t}}(\hat{S}) \leq b_{\hat{t}}^{\mathcal{M}} + r_{\hat{t}}(S'). \tag{20}$$

Let $\overline{\mathcal{W}} = \{\overline{w}_1, \overline{w}_2, \ldots, \overline{w}_l\}$ be the set of workers that have been matched to task $\hat{t}$ at some point during the course of the execution of Algorithm 2. The corresponding coverage sets of these workers are $\overline{\mathcal{C}}_{\hat{t}} = \{C_{\hat{t}}^{\overline{w}_i} : 1 \leq i \leq l\}$, and the rewards offered to these workers by task $\hat{t}$ are given as $\overline{\mathcal{R}}_{\hat{t}} = \{r_{\hat{t}}(\overline{w}_i) : 1 \leq i \leq l\}$.

Note also that the algorithm attempts to match a single worker-task pair at a time, and if a worker w is first matched with a task t (line 9) and then removed from $\mathcal{M}(t)$ at some point (lines 15-18), the algorithm will never attempt to match him with task t afterwards, instead it will try to match him with other tasks which come after task t in his preference list

Algorithm 3: WorkerSelection (t, w, $\mathcal{M}$)

Input: t: Task evaluating worker w
w: Candidate worker
$\mathcal{M}$: Current matching

1 $x_t(w) = 1$
2 **foreach** $p \in C_t^w$ **do**
3 $\quad z_t(p, w) = 1 - \sum_{w' \in H_t} z_t(p, w')$
4 $\quad \eta_t(w) = \eta_t(w) + z_t(p, w) v_t(p)$
5 **end**
6 $\eta_t(w) = \eta_t(w) \times b_t / r_t(w)$
7 **if** $\eta_t(w) \leq 2 \times \sum_{p \in P(t)} \sum_{w' \in H_t} z_t(p, w') v_t(p)$ **then**
8 $\quad$ **return** $\{w\}$
9 **end**
10 insert w to H_t by maintaining the order, i.e., $H_t = \{\hat{w}_1, \hat{w}_2, .., \hat{w}_q\}$ s.t. $\eta_t(\hat{w}_i) \geq \eta_t(\hat{w}_{i+1})$, $\forall i < q$
11 $k \leftarrow \max\{k' \leq |H_t| : \gamma = \sum_{i=1}^{k-1} r_t(w) < b_t\}$
12 $b_t^{\mathcal{M}} = b_t - \gamma$
13 $w' \leftarrow k$th worker in H_t
14 $\beta = \min\{b_t^{\mathcal{M}} / r_t(w'), 1\}$
15 **foreach** $p \in C_t^{w'}$ **do**
16 $\quad z_t(p, w') = \frac{\beta}{x_t(w')} z_t(p, w')$
17 **end**
18 $\mathcal{R} = \emptyset$
19 **if** $x_t(w') = 1$ *and* $\beta < 1$ **then**
20 $\quad \mathcal{R} \leftarrow \{w'\}$
21 **end**
22 $x_t(w') = \beta$
23 **for** $i \leftarrow |H_t|$ *down to* $k+1$ **do**
24 $\quad \hat{w} \leftarrow i$th worker in H_t
25 $\quad H_t$.remove($\hat{w}$)
26 $\quad$ **if** $x_t(\hat{w}) = 1$ **then**
27 $\quad\quad \mathcal{R} \leftarrow \mathcal{R} \cup \hat{w}$
28 $\quad$ **end**
29 **end**
30 **return** $\mathcal{R}$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2021.3065688, IEEE Transactions on Vehicular Technology

L_w. Thus, from a task's perspective, say task $\hat{t}$, this is exactly the same problem with the OBMC problem, as we have a collection $\overline{\mathcal{C}}_{\hat{t}}$ of sets with associated costs $\overline{\mathcal{R}}_{\hat{t}}$ that arrive one at a time, and that cannot be later included to the solution $\mathcal{M}(\hat{t})$ after they are discarded, and the goal of task $\hat{t}$ is also to maximize the weighted coverage within the budget. In fact, we can map the two problems to each other as follows:

$$\begin{aligned}\mathcal{U} &\longleftrightarrow P(\hat{t}),\\ \mathcal{S} &\longleftrightarrow \overline{\mathcal{C}}_{\hat{t}},\\ S_i &\longleftrightarrow C_{\hat{t}}^{\overline{w}_i},\\ c_i &\longleftrightarrow r_{\hat{t}}(\overline{w}_i),\\ B &\longleftrightarrow b_{\hat{t}}.\end{aligned}$$

For this reason, Algorithm 2 runs the adapted version of the OBMC algorithm (i.e., Algorithm 3) as a subroutine (line 11) to decide which workers to keep in the worker set of a task after the algorithm attempts to assign another worker to her. Then, by Theorem 2, we have that

$$U_{\hat{t}}(\mathcal{S}_{best}) < \frac{4 \times U_{\hat{t}}(\mathcal{M}(\hat{t}))}{1-\rho_{\hat{t}}} \tag{21}$$

where $\mathcal{S}_{best} \subseteq \overline{W}$ is the best set that could be assigned to task $\hat{t}$ providing the highest total weighted coverage within the budget constraint (i.e., an optimal solution of the corresponding online budgeted maximum coverage problem), and $\rho_{\hat{t}} = \max_{w \in \overline{\mathcal{W}}} \frac{r_{\hat{t}}(w)}{b_{\hat{t}}}$.

Note that $\hat{S}$ cannot include a worker that is not in $\overline{\mathcal{W}}$, thus we have

$$\left(\hat{S} \cup (\mathcal{M}(\hat{t}) \setminus S')\right) \subseteq \overline{\mathcal{W}}. \tag{22}$$

That is because, by Definition 1, all workers in $\hat{S}$ must be preferring task $\hat{t}$ to the tasks they are currently assigned. However, if a worker w is currently matched with task t', the algorithm should have attempted to assign him all the other tasks that precede task t' in his preference list. Then, if worker w prefers task $\hat{t}$ to task t', which means that task $\hat{t}$ also precedes task t' in his preference list, we must have $w \in \overline{\mathcal{W}}$.

Due to (22) and the fact that $\mathcal{S}_{best}$ is the best feasible set in $\overline{\mathcal{W}}$ for task $\hat{t}$, we have

$$U_{\hat{t}}(\mathcal{S}_{best}) \geq U_{\hat{t}}(\hat{S} \cup (\mathcal{M}(\hat{t}) \setminus S')). \tag{23}$$

Then, combining the inequalities (20), (21) and (23), we get

$$\frac{4 \times U_{\hat{t}}(\mathcal{M}(\hat{t}))}{1-\rho_{\hat{t}}} > \frac{4 \times U_{\hat{t}}(\mathcal{M}(\hat{t}))}{1-\rho} \tag{24}$$

which is a contradiction as $\rho \geq \rho_{\hat{t}}$. □

2) Proportional Reward Scheme: We propose Algorithm 4 for the proportional reward scheme. It directly runs Algorithm 3 (line 18) for task t until the main algorithm attempts to assign her a worker w that satisfies $r_t(w) \geq 0.2 \times b_t$. When this happens, the algorithm finalizes the assignment of worker w to task t (i.e., in the end, they will be matched to each other), and updates the budget of task t (line 5) and coverage sets of workers (lines 6-8) to reflect the fact that a certain proportion of task t's budget is not available anymore, and that the utility of the other workers should be computed considering only the PoIs that are not covered by worker w. It also attempts to re-assign the previous worker set of task t to her considering the modified budget and coverage sets (lines 11-15). In the subsequent iterations in which the main algorithm attempts to assign another worker to task t, since $\mathcal{A}_t$ is previously set to *true* (line 2), the algorithm continues to run Algorithm 3 with the modified budget of task t and coverage sets of workers (line 18).

Algorithm 4: WorkerSelectionProportional (t,w,$\mathcal{M}$)

Input: t: Task evaluating worker w
w: Candidate worker
$\mathcal{M}$: Current matching

1 **if** $\mathcal{A}_t$ *is false and* $r_t(w) \geq 0.2 \times b_t$ **then**
2 $\quad \mathcal{A}_t \leftarrow$ *true*
3 $\quad \mu \leftarrow \mathcal{M}(t)$
4 $\quad \mathcal{M}(t) \leftarrow \{w\}$
5 $\quad b_t = b_t - r_t(w)$
6 $\quad$ **foreach** $w' \in \mathcal{W} \setminus \{w\}$ **do**
7 $\quad\quad C_t^{w'} \leftarrow C_t^{w'} \setminus C_t^{w}$
8 $\quad$ **end**
9 $\quad H_t \leftarrow \emptyset$
10 $\quad \mathcal{R} \leftarrow \emptyset$
11 $\quad$ **foreach** $w' \in \mu$ **do**
12 $\quad\quad \mathcal{M}(t) = \mathcal{M}(t) \cup \{w'\}$
13 $\quad\quad \eta_t(w') = 0$
14 $\quad\quad R \leftarrow R \cup$ WorkerSelection$(t, w', \mathcal{M})$
15 $\quad$ **end**
16 $\quad$ **return** $\mathcal{R}$
17 **else**
18 $\quad$ **return** *WorkerSelection*$(t, w, \mathcal{M})$
19 **end**

Theorem 4. *Algorithm 2 always produces* 5*-stable matchings in the presence of a proportional reward scheme.*

Proof. We prove it by contradiction. Let $\mathcal{M}$ be the returned matching by the algorithm, and $\langle \hat{S}, \hat{t} \rangle$ be an unhappy coalition in $\mathcal{M}$ that, for some $S' \subseteq \mathcal{M}(\hat{t})$, satisfies

$$\begin{aligned} U_{\hat{t}}(\hat{S} \cup (\mathcal{M}(\hat{t}) \setminus S')) &> 5 \times U_{\hat{t}}(\mathcal{M}(\hat{t}))\\ \text{and } r_{\hat{t}}(\hat{S}) &\leq b_{\hat{t}}^{\mathcal{M}} + r_{\hat{t}}(S'). \end{aligned} \tag{25}$$

Thus, $\langle \hat{S}, \hat{t} \rangle$ prevents $\mathcal{M}$ from being a 5-stable matching according to Definition 3.

If $\mathcal{A}_{\hat{t}}$ is true in the end, then $\mathcal{M}(\hat{t})$ includes a worker w such that

$$r_{\hat{t}}(w) \geq 0.2 \times b_{\hat{t}}. \tag{26}$$

Since the utility $U_{\hat{t}}(w)$ of worker w is proportional to his reward $r_{\hat{t}}(w)$, we have

$$U_{\hat{t}}(\mathcal{M}(\hat{t})) \geq U_{\hat{t}}(w) = \theta_t \times r_{\hat{t}}(w) \tag{27}$$

Also, given the budget limit of task $\hat{t}$, the total weighted coverage that the best feasible worker set can provide for task $\hat{t}$ is at most

$$U_{max} = \theta_t \times b_{\hat{t}}. \tag{28}$$

Then, by (26), (27) and (28), we get

$$\begin{aligned} 5 \times U_{\hat{t}}(\mathcal{M}(\hat{t})) &\geq U_{max} \\ &\geq U_{\hat{t}}(\hat{S} \cup (\mathcal{M}(\hat{t}) \setminus S')) \end{aligned} \tag{29}$$

which contradicts (25).

On the other hand, if $\mathcal{A}_{\hat{t}}$ is false in the end, then only Algorithm 3 has been run for task t, similar to the case with the general reward scheme, thus the inequality (21) must hold here, as well. Since the inside of the if block in Algorithm 4 is never executed, we have that $r_{\hat{t}}(w) < 0.2 \times b_{\hat{t}}$, for all $w \in \overline{\mathcal{W}}$. Then, (21) becomes

$$U_{\hat{t}}(\mathcal{S}_{best}) < 5 \times U_{\hat{t}}(\mathcal{M}(\hat{t})) \tag{30}$$

Following the same steps ((22) and (23)) in the proof of Theorem 3, we obtain

$$5 \times U_{\hat{t}}(\mathcal{M}(\hat{t})) > 5 \times U_{\hat{t}}(\mathcal{M}(\hat{t})) \tag{31}$$

which is also false and completes the proof. □

C. Feasibility, Rationality and Efficiency

We lastly show that the proposed algorithms always produce individually rational and feasible matchings, and analyze their asymptotic running times.

Theorem 5. *Algorithm 2 always produces individually rational and feasible matchings.*

Proof. Note that a worker can only get matched with a task in his preference list (line 7), and matching with any of the tasks in his preference list is profitable for him since those that are not so are removed from his preference list in Algorithm 1 (line 3). Thus, we conclude that the produced matchings are individually rational. It is clear from lines 9 & 16 of Algorithm 2 that the produced matchings are feasible in terms of mutual partnership. As for the budget feasibility, when the reward scheme is general, Algorithm 3 returns (line 11) the set of the least efficient workers that need to be removed from the worker set of task t to stay within the budget constraint b_t, which are then actually removed from the worker set of task t in lines 15-18 of Algorithm 2. When the reward scheme is proportional, Algorithm 4 either only runs Algorithm 3 (line 18), or executes the inside of the if block beginning in line 1 at most once for each task t, where the budget of task t is decreased (line 5) by the reward amount that will be paid to the accepted worker w. After that, it always runs Algorithm 3 for task t. Therefore, the produced matchings are also feasible. □

Running time. The initialization (i.e., running Algorithm 1) takes $\mathcal{O}(knm + mn\log(n))$, where the latter term is due to sorting $\mathcal{T}$ for each worker. In Algorithm 2, each worker w is pushed onto the stack at most $|L_w| \leq \mathcal{T}$ times, thus the while loop iterates at most nm times. The costliest operations in the while loop for the *general* and *proportional* reward schemes are running Algorithm 3 (line 11) and Algorithm 4 (line 13), respectively. Algorithm 3 runs in $\mathcal{O}(km)$ time. Since the inside of the if block in Algorithm 4 will be run at most once and hence Algorithm 3 will be called at most 2 times for each worker-task pair (from lines 14 and 18 in Algorithm 4), the amortized cost of Algorithm 4 also becomes $\mathcal{O}(km)$. Therefore, the worst-case running time of our approximation algorithm is $\mathcal{O}(knm^2 + mn\log(n))$ regardless of the reward scheme.

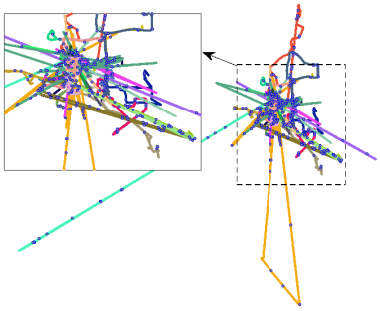


Fig. 2: Trajectories of the workers in the KAIST data set (circles denote the PoIs).

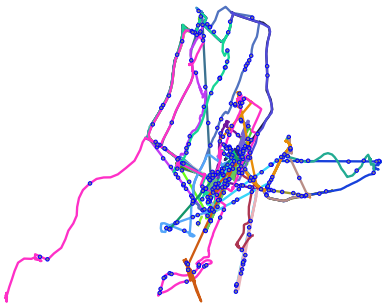


Fig. 3: Trajectories of the workers in the NYC data set (circles denote the PoIs).

V. Simulation Results

In this section, we present an extensive evaluation of the proposed algorithms in MCS systems with both general and proportional reward schemes.

A. Settings

Similar to [11], we utilize two real data sets [45], [46] that consist of the trajectories of 39 and 92 participants from New York City (NYC) and the campus of the Korea Advanced Institute of Science and Technology (KAIST), respectively. We create 300 PoIs at random locations that are on the trajectory (i.e., within 50 meters) of at least one participant. Fig. 2 and 3 show the trajectories in the data sets and an example of PoI distribution.

We let the trajectories in the data sets to be the trajectories of the workers in our system. To look at the impact of the number

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2021.3065688, IEEE Transactions on Vehicular Technology

of workers, we use random-sampling to obtain a worker set of certain size. According to the experiment requirements, we create n tasks whose budgets are assigned randomly from [10, 100]. In order to determine the PoI list of a task t, we first select a random number s from [1, 25]. Then, we randomly insert s of the all PoIs to $P(t)$ after assigning a random weight value from (0, 1]. Since we utilize deterministic trajectories, the lists of PoIs to be visited by workers and the tasks they can complete are known in advance. The assignment of the rewards for different reward schemes is made as follows:

- **General reward scheme:** For each worker-task pair (w, t), we assign the reward $r_t(w)$ randomly from the range $[0.05 \times b_t, 0.95 \times b_t]$. If $C_t^w = \emptyset$, we set $r_t(w) = 0$.
- **Proportional reward scheme:** For each worker-task pair (w, t), the reward is set as

$$r_t(w) = b_t \times \frac{\sum_{p \in C_t^w} v_t(p)}{\sum_{p \in P(t)} v_t(p)}. \tag{32}$$

Since the rewards are already determined based on the randomly assigned budget values, we let $c_t(w) = 0$ for all worker-task pairs (w, t)[2].

In the simulations, we let CSTA$_G$ and CSTA$_P$ denote the execution of the proposed *C*overage-aware *S*table *T*ask *A*ssignment algorithm with general and proportional reward schemes, respectively. We compare the performance of these algorithms with that of Maximum Coverage Quality Assignment (or *MCQA*) algorithm proposed in [11] and *Greedy* algorithm proposed in [12] for the problem of finding the worker set with the maximum total weighted coverage over a given set of PoIs. They both are originally proposed for the MCS systems with only a single task requester, m workers, and k PoIs. The MCQA algorithm has an approximation ratio of $(1-1/e)$ for the aforementioned optimization problem and a time complexity of $\mathcal{O}(kn^5)$. On the other hand, the Greedy algorithm does not have a theoretical performance guarantee and runs in $\mathcal{O}(knm^2)$. We adapt them to our settings with multiple task requesters as follows. For each task t in the system, we first find the set S of workers that, among all the tasks in the system, prefer task t the most. Then, we separately run the MCQA/Greedy algorithm for each such (t, S) pair with $P(t)$ being the set of PoIs, and finalize the assignments made in each run. Lastly, for each worker w that is still unmatched, we traverse his preference list L_w from the beginning, and match him with the first task that benefits from hiring him (i.e., worker w increases the coverage quality of task t) and has sufficient budget to do so. These adapted versions are denoted by *MCQA** and *Greedy** in the simulations.

B. Performance Metrics

Here, we introduce the performance metrics that will be used in the evaluation of the results.

- *Stability success ratio (%)*: This metric shows how often an algorithm achieves the best known upper-bound in terms of stability in different settings. Specifically, let $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_{100}$ be the matchings produced by an algorithm $\mathcal{A}$ in 100 consecutive runs on different MCS instances. Also let

$$s(\mathcal{M}_i, \alpha) = \begin{cases} 1, & \text{if } \mathcal{M}_i \text{ is an } \alpha\text{-stable matching,} \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

Then, the stability success ratio of $\mathcal{A}$ is calculated by

$$\sum_{i=1}^{100} s(\mathcal{M}_i, \frac{4}{1-\rho}) \tag{34}$$

for the MCS systems with a general reward scheme, and by

$$\sum_{i=1}^{100} s(\mathcal{M}_i, 5) \tag{35}$$

for the MCS systems with a proportional reward scheme.

- *User happiness (%)*: This quantifies the user happiness based on the stability of the matching as follows:

$$100 \times \left(1 - \frac{\text{\# of coalitionally unhappy pairs}}{\text{\# of qualified pairs}}\right) \tag{36}$$

- *Stability (σ)*: This is the value of the objective function defined in (17), and indicates that the produced matching is $(1/\sigma)$-stable. If there is not any unhappy coalition in the matching, then $\sigma = 1$ (by definition of δ_t).

- *Average coverage quality (%)*: This is the average weighted coverage that the produced matching, $\mathcal{M}$, provides for the tasks, or formally:

$$\frac{100}{n} \times \sum_{t \in \mathcal{T}} \frac{U_t(\mathcal{M}(t))}{\sum_{p \in P(t)} v_t(p)}. \tag{37}$$

- *Running time*: In order to show the scalability of the algorithms, we also present their running times with increasing number of workers/tasks/PoIs on an Intel core i7 processor with 16 GB memory and 2.5 GHz speed.

Lastly, we note that all results provided in this section are the average of the results obtained in 100 runs (each with a different MCS instance).

C. Results

We first analyze the results for the KAIST data set. Fig. 4 & 5 show the impact of the number of tasks n on the performance of the algorithms with general and proportional reward schemes, respectively. First, note that the performances of the MCQA* and Greedy* algorithms in terms of stability (Fig. 4b and Fig. 5b) deteriorate significantly as n increases. This is due to the fact that these algorithms do not consider the system as a whole, and aim to maximize the coverage for each task separately. However, since they optimize the assignments for individual tasks extensively (which, in turn, increases their running time significantly as can be seen in Fig. 12), they outperform the other algorithms when n is small in terms of user happiness (Fig. 4c and Fig. 5c) and average coverage quality (Fig. 5d).

[2]Introducing extra, random cost values naturally reduces the number of qualified pairs and consequently the average coverage quality, but it does not have a notable effect on the relative performance of the algorithms.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2021.3065688, IEEE Transactions on Vehicular Technology

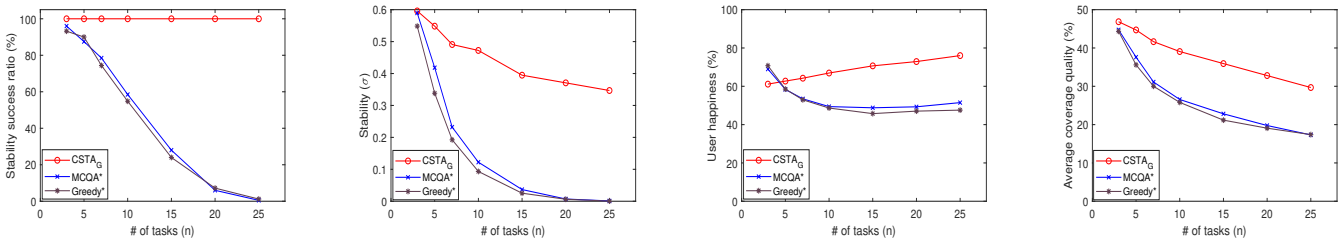


Fig. 4: General setting: performance comparison against varying number of tasks in the KAIST data set ($m = 92$).

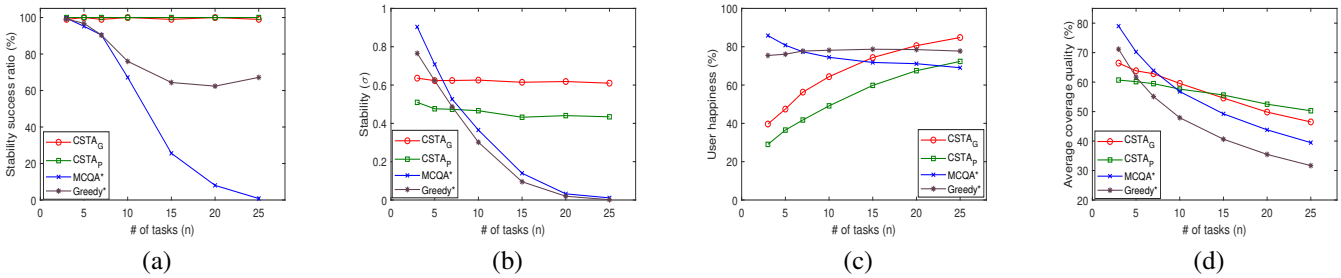


Fig. 5: Proportional setting: performance comparison against varying number of tasks in the KAIST data set ($m = 92$).

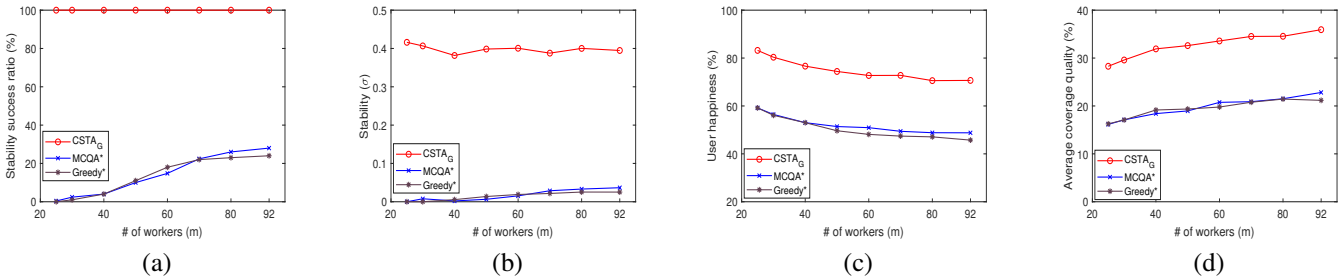


Fig. 6: General setting: performance comparison against varying number of workers in the KAIST data set ($n = 15$).

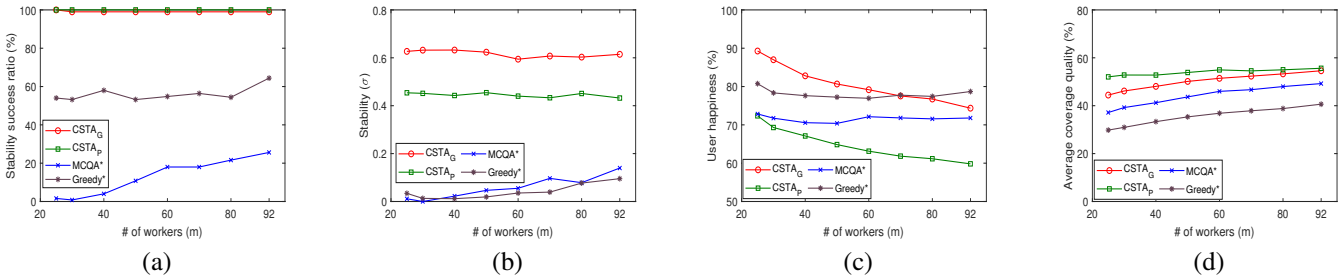
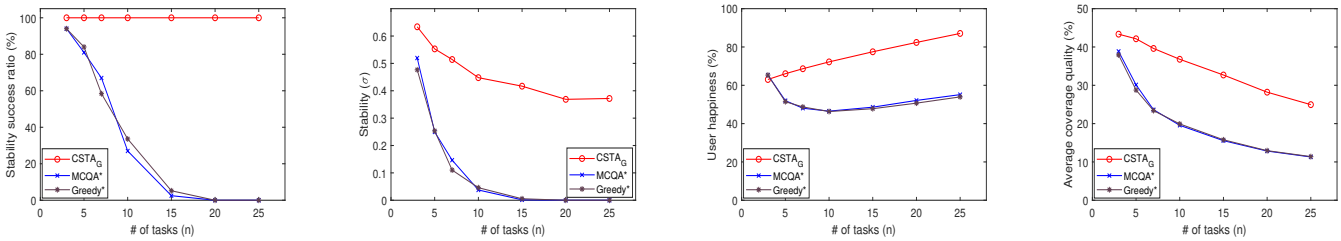


Fig. 7: Proportional setting: performance comparison against varying number of workers in the KAIST data set ($n = 15$).

In terms of stability success ratio, the CSTA$_G$ and CSTA$_P$ algorithms produce perfect task assignments in the general and proportional settings, respectively, as expected (due to Theorem 3 & 4), and vastly outperform the other algorithms. We see that the CSTA$_G$ algorithm occasionally fails to produce perfect assignments in terms of stability success ratio in the proportional setting, which indicates that assigning a task t with the first worker w such that $r_t(w) \geq 0.2 \times b_t$ (lines 1-17 in the CSTA$_P$ algorithm) is required to achieve 5-stable matchings. Yet, this comes with a trade-off as the CSTA$_P$ algorithm yields significantly lower stability scores (σ) compared to the CSTA$_G$ algorithm as seen in Fig. 5b. Since the Greedy* algorithm selects workers according to the ratio of how much utility they will bring for the tasks to the reward they will be paid, its performance is much better than the MCQA* algorithm in the proportional setting where the value of the proposed reward per utility is constant for all workers.

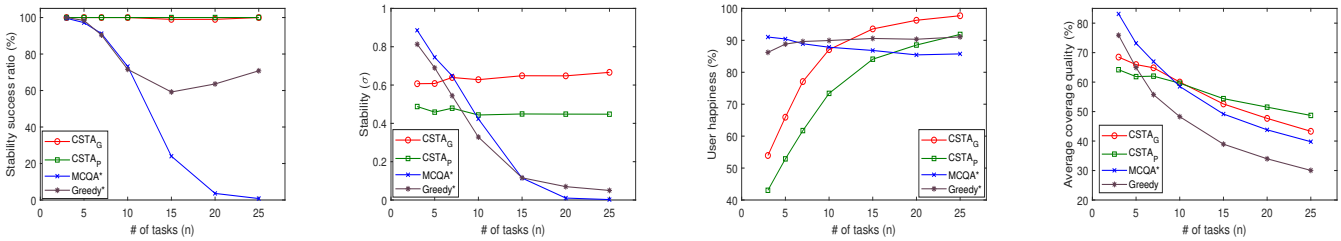
In the average coverage quality graphs (Fig. 4d and 5d), we see that the average coverage decreases for all algorithms with increasing n as there will be fewer workers assigned to each task. We also see that the coverage scores in the proportional setting are remarkably larger than those in the general setting mainly because of the discrepancy between reward and utility

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2021.3065688, IEEE Transactions on Vehicular Technology



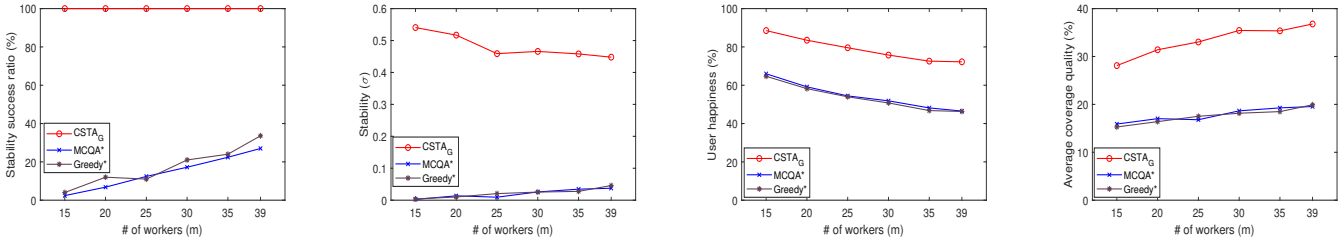
(a) (b) (c) (d)

Fig. 8: General setting: performance comparison against varying number of tasks in the NYC data set ($m = 39$).



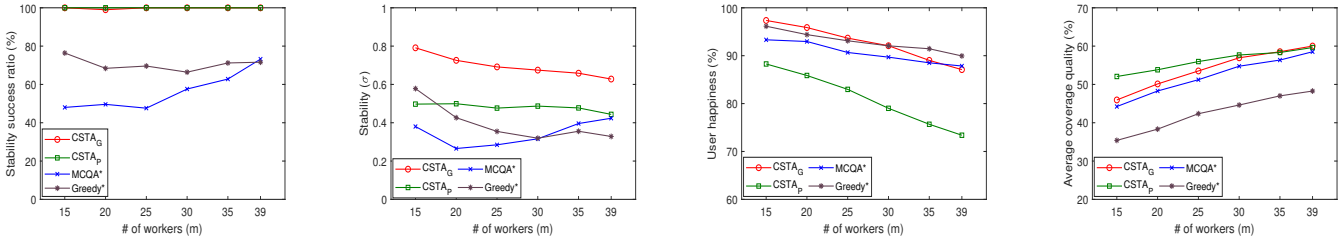
(a) (b) (c) (d)

Fig. 9: Proportional setting: performance comparison against varying number of tasks in the NYC data set ($m = 39$).



(a) (b) (c) (d)

Fig. 10: General setting: performance comparison against varying number of workers in the NYC data set ($n = 10$).



(a) (b) (c) (d)

Fig. 11: Proportional setting: performance comparison against varying number of workers in the NYC data set ($n = 10$).

values in the latter setting (i.e., a high reward does not indicate a high utility for tasks, unlike the proportional setting). It is also noteworthy that in terms of coverage, the proposed algorithms mostly outperform the MCQA* and Greedy* algorithms, whose sole objective is to maximize the coverage. This demonstrates that taking user preferences into account does not necessarily yield less efficient assignments in terms of system-level utility metrics such as coverage.

Next, we look at the performance of the algorithms with varying number of workers in Fig. 6 & 7. Except for the user happiness results, we observe that *increasing* the number of workers m has a similar impact on the performance of the MCQA* algorithm with *decreasing* the number of tasks n. This is because both changes result in a smaller ratio of n to m (i.e., task scarcity), which alleviates the deficiency of the MCQA* algorithm in handling multi-task assignments. This is also mostly true for the Greedy* algorithm, however its performance in terms of stability success ratio in the proportional setting is more stable. We note that the changes in the number of tasks or workers do not have a significant impact on the stability and stability success ratio scores of the proposed algorithms in the proportional setting as seen in Fig. 5a-b and 7a-b. Another remarkable point is that the MCQA* and Greedy* algorithms have almost identical

performance in the general setting with varying n and m values, yet their performance in the proportional setting is quite different. Specifically, in terms of stability success ratio and user happiness, the Greedy* algorithm mostly outperforms the MCQA* algorithm, while it is the opposite in terms of stability and average coverage quality.

Fig. 6 & 7 show that the CSTA$_G$ algorithm always outperforms the MCQA* algorithm in terms of user happiness (by up to 25%) regardless of the number of workers, but it is slightly outperformed by the Greedy* algorithm when m is larger than 70 in the proportional setting, and that the performance of the proposed algorithms mostly degrades as m increases. We observe that all algorithms achieve higher coverage scores with increasing m values, which is naturally the opposite of what we see with increasing n values. This is because if there are more workers per task in the system, the competition between tasks will be less severe, and each task will be assigned to a higher number of workers, on average. However, the budget constraints of the tasks limit the number of workers that can be assigned to them, hence we start to see a smaller or no increase in coverage after some point, especially in the proportional setting. We also note that the stability of the matchings produced by the proposed algorithms is significantly higher than the theoretical upper-bound (0.2) in the proportional settings (Fig. 5b & 7b). Besides, Fig. 6b & 7b demonstrate that our algorithms always significantly outperform the MCQA* and Greedy* algorithm in terms of σ. The difference in σ is especially big (up to 0.6) when the ratio of n to m is larger.

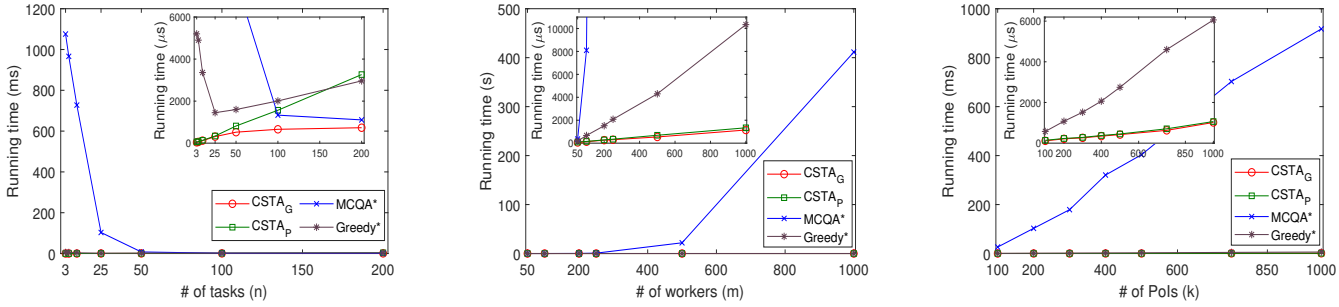
In order to demonstrate that the results provided above are not specific to a data set, we also examine the performance of the algorithms in the NYC data set in Fig. 8, 9, 10 & 11. The proposed algorithms in general perform better than the benchmark algorithm as in the KAIST data set. The results in both data sets are similar, thus the majority of our comments above for the KAIST data set also apply to the results for the NYC data set. However, there are some differences in results we see in the NYC data set. The first significant difference can be seen between Fig. 7c and Fig. 11c. Here, we see that increasing the number of workers continues to improve the achieved coverage in the NYC data set, while there is mostly little to no improvement in the KAIST data set. This is primarily because of the limited number of workers (39) available in the NYC data set. That is, since the tasks still have budget for more workers, adding new workers to the system simply expands the coverage. This can also be partially observed in Fig. 7 up until $m = 60$. Another noteworthy difference is in the user happiness results in proportional setting. All algorithms accomplish better user happiness scores (up to 20%) in the NYC data set compared to those in KAIST data set (i.e., Fig. 5b vs. Fig. 9b and Fig. 7b vs. Fig. 11b). This might be because the trajectories of the workers in the NYC data set are more dispersed than those in the KAIST data set (see Fig. 2 & 3), which, in turn, reduces the overall competition between the tasks as such a difference in the trajectories implies that the PoIs covered by the workers differ more in the NYC data set, and the workers are hence favored by different tasks.

Lastly, we compare the running times of the algorithms in Fig. 12. In order to show the scalability of the algorithms for large numbers of tasks, workers and PoIs, we generated a synthetic data set in a 3,000 m $\times$ 3,000 m area with k randomly located PoIs, m workers whose trajectories are created using the random-walk mobility model (as in [11]) for 2,000 meters (with a direction change at every 200 meters), and n tasks whose PoI sets and budgets are determined exactly as in the real data sets. Since the proportional setting allows us to compare the running times of all algorithms, the rewards are assigned using the proportional reward mechanism (the running times of the MCQA*, Greedy* and CSTA$_G$ algorithms in proportional setting are similar to their running times in the general setting).

Recall that in the MCQA* algorithm, the original MCQA algorithm is run separately for each task t and the set Q_t of workers that prefer task t the most, and the time complexity of each such run is $\mathcal{O}(\hat{k}\hat{m}^5)$, where $\hat{k} = |P(t)|$ and $\hat{m} = |Q_t|$. Since fewer tasks in the system means that for each task t, there will be more workers that prefer t the most (i.e., a larger $|Q_t|$), the running time of the MCQA* algorithm increases when n decreases (Fig. 12a) or m increases (Fig. 12b). The time complexity of the original Greedy algorithm is $\mathcal{O}(knm^2)$, and its running time decreases with increasing n values up until $n = 25$ due to the same reason (i.e., fewer workers per task). After this point, the second phase of the Greedy* algorithm, which also has a time complexity of $\mathcal{O}(knm^2)$ and is where the algorithm tries to match the workers that could not get matched with any tasks during the first phase as proposed in our adaptation, starts to dominate the running time and we begin to see a linear growth. In these figures, we also see that the running times of the CSTA$_G$ and CSTA$_P$ algorithms are mostly a few orders of magnitude smaller than that of the MCQA* algorithm. This is simply because of the superior time complexity of these algorithms: $\mathcal{O}(knm^2 + mn\log(n))$. They also run significantly faster than the Greedy* algorithm. Note that while the complexity of proposed algorithms will be the same as the Greedy* algorithm, because $\log(n) \ll km$ for most values used in practice, their actual running times are less than that of the Greedy* algorithm. This is because the preference list of each worker in the proposed algorithms contains only a limited number of tasks as a rational worker will accept only the tasks that request data from some of the POIs on his trajectory (line 3 of Algorithm 1). So, the number of times workers are pushed onto the stack is generally much smaller than $n \times m$, making $\mathcal{O}(knm^2)$ not tight. Finally, we note that the running times of all algorithms increase linearly with the increasing number of PoIs k as seen in Fig. 12c, which is in accordance with the influence of k in their asymptotic running times.

VI. Conclusion

In this paper, we study the problem of finding stable multi-task assignments with weighted coverage-based utility functions in a budget-constrained and opportunistic mobile crowdsensing scenario. We first define the stability (or user happiness) conditions within this scenario, and point out the



(a) (b) (c)

Fig. 12: Running times of the algorithms with varying number of (a) tasks ($m = 200$, $k = 300$); (b) workers ($n = 20$, $k = 300$); and (c) PoIs ($n = 20$, $m = 200$) with the proportional reward scheme in the synthetic data set.

hardness of the problem and nonexistence of optimal solutions in some cases. We then present two approximation algorithms and derive their approximation ratios in different settings. Finally, we provide an extensive evaluation of the proposed algorithms, which demonstrates that they largely outperform the considered benchmark algorithms in terms of both user happiness and coverage quality while having significantly smaller running times (up to 4 orders of magnitude). In our future work, we will address the coverage-aware stable task assignment problem in an online scenario where both workers and tasks can arrive and leave at any time, and when worker trajectories thus the POIs that will be visited by workers are uncertain.

ACKNOWLEDGMENT

This material is based upon work supported by the U.S. National Science Foundation (NSF) under Grant CNS1647217.

REFERENCES

[1] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2419–2465, 2019. [Online]. Available: https://doi.org/10.1109/COMST.2019.2914030

[2] Y. Chon, Y. Kim, and H. Cha, "Autonomous place naming system using opportunistic crowdsensing and knowledge from crowdsourcing," in *The 12th International Conference on Information Processing in Sensor Networks (co-located with CPS Week 2013), IPSN 2013, Philadelphia, PA, USA, April 8-11, 2013*, 2013, pp. 19–30.

[3] W. Gong, B. Zhang, and C. Li, "Location-based online task assignment and path planning for mobile crowdsensing," *IEEE Trans. Vehicular Technology*, vol. 68, no. 2, pp. 1772–1783, 2019.

[4] M. Karaliopoulos, O. Telelis, and I. Koutsopoulos, "User recruitment for mobile crowdsensing over opportunistic networks," in *2015 IEEE Conference on Computer Communications, INFOCOM 2015, Kowloon, Hong Kong, April 26 - May 1, 2015*, 2015, pp. 2254–2262.

[5] B. Guo, Y. Liu, W. Wu, Z. Yu, and Q. Han, "Activecrowd: A framework for optimized multitask allocation in mobile crowdsensing systems," *IEEE Trans. Human-Machine Systems*, vol. 47, no. 3, pp. 392–403, 2017. [Online]. Available: https://doi.org/10.1109/THMS.2016.2599489

[6] M. Xiao, J. Wu, L. Huang, Y. Wang, and C. Liu, "Multi-task assignment for crowdsensing in mobile social networks," in *2015 IEEE Conference on Computer Communications, INFOCOM 2015, Kowloon, Hong Kong, April 26 - May 1, 2015*, 2015, pp. 2227–2235.

[7] H. Xiong, D. Zhang, G. Chen, L. Wang, and V. Gauthier, "Crowdtasker: Maximizing coverage quality in piggyback crowdsensing under budget constraint," in *2015 IEEE International Conference on Pervasive Computing and Communications, PerCom 2015, St. Louis, MO, USA, 23-27 March, 2015*, 2015, pp. 55–62.

[8] J. Wang, J. Tang, G. Xue, and D. Yang, "Towards energy-efficient task scheduling on smartphones in mobile crowd sensing systems," *Computer Networks*, vol. 115, pp. 100 – 109, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128616304145

[9] Y. Liu, B. Guo, Y. Wang, W. Wu, Z. Yu, and D. Zhang, "Taskme: multi-task allocation in mobile crowd sensing," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016, Heidelberg, Germany, September 12-16, 2016*, 2016, pp. 403–414.

[10] Z. Feng, Y. Zhu, Q. Zhang, L. M. Ni, and A. V. Vasilakos, "TRAC: truthful auction for location-aware collaborative sensing in mobile crowdsourcing," in *2014 IEEE Conference on Computer Communications, INFOCOM 2014, Toronto, Canada, April 27 - May 2, 2014*, 2014, pp. 1231–1239. [Online]. Available: https://doi.org/10.1109/INFOCOM.2014.6848055

[11] M. Zhang, P. Yang, C. Tian, S. Tang, X. Gao, B. Wang, and F. Xiao, "Quality-aware sensing coverage in budget-constrained mobile crowdsensing networks," *IEEE Trans. Vehicular Technology*, vol. 65, no. 9, pp. 7698–7707, 2016.

[12] Jiaoyan Chen and Jingsen Yang, "Maximizing Coverage Quality with Budget Constrained in Mobile Crowd-Sensing Network for Environmental Monitoring Applications," *Sensors*, vol. 19, no. 10, p. 2399, 2019.

[13] Z. Zheng, F. Wu, X. Gao, H. Zhu, S. Tang, and G. Chen, "A budget feasible incentive mechanism for weighted coverage maximization in mobile crowdsensing," *IEEE Trans. Mob. Comput.*, vol. 16, no. 9, pp. 2392–2407, 2017. [Online]. Available: https://doi.org/10.1109/TMC.2016.2632721

[14] F. Yucel and E. Bulut, "User satisfaction aware maximum utility task assignment in mobile crowdsensing," *Computer Networks*, vol. 172, p. 107156, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128619311697

[15] D. Gale and L. Shapley, "College admissions and stability of marriage. american mathematicas monthly, 69, 9-15," 1962.

[16] F. Yucel and M. Yuksel and E. Bulut, "QoS-based Budget Constrained Stable Task Assignment in Mobile Crowdsensing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[17] Y. Chen and X. Yin, "Stable job assignment for crowdsourcing," in *IEEE Global Communications Conference*, 2017, pp. 1–6.

[18] M. Abououf, S. Singh, H. Otrok, R. Mizouni, and A. Ouali, "Gale-shapley matching game selection - A framework for user satisfaction," *IEEE Access*, vol. 7, pp. 3694–3703, 2019.

[19] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing, and X. Mao, "Incentives for mobile crowd sensing: A survey," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 54–67, 2016. [Online]. Available: https://doi.org/10.1109/COMST.2015.2415528

[20] L. G. Jaimes, I. J. Vergara-Laurens, and A. Raij, "A survey of incentive techniques for mobile crowd sensing," *IEEE Internet of Things Journal*, vol. 2, no. 5, pp. 370–380, 2015. [Online]. Available: https://doi.org/10.1109/JIOT.2015.2409151

[21] Y. Chen, P. Lv, D. Guo, T. Zhou, and M. Xu, "A survey on task and participant matching in mobile crowd sensing," *J. Comput. Sci. Technol.*, vol. 33, no. 4, pp. 768–791, 2018. [Online]. Available: https://doi.org/10.1007/s11390-018-1855-y

[22] W. Gong, B. Zhang, and C. Li, "Task assignment in mobile crowdsensing: Present and future directions," *IEEE Network*, vol. 32, no. 4, pp. 100–107, 2018. [Online]. Available: https://doi.org/10.1109/MNET.2018.1700331

[23] Y. Wen, J. Shi, Q. Zhang, X. Tian, Z. Huang, H. Yu, Y. Cheng, and X. Shen, "Quality-driven auction-based incentive mechanism for mobile crowd sensing," *IEEE Trans. Vehicular Technology*, vol. 64, no. 9, pp. 4203–4214, 2015.

[24] H. Gao, C. H. Liu, J. Tang, D. Yang, P. Hui, and W. Wang, "Online quality-aware incentive mechanism for mobile crowd sensing with extra bonus," *IEEE Trans. Mob. Comput.*, vol. 18, no. 11, pp. 2589–2603, 2019. [Online]. Available: https://doi.org/10.1109/TMC.2018.2877459

[25] H. Wang, S. Guo, J. Cao, and M. Guo, "Melody: A long-term dynamic quality-aware incentive mechanism for crowdsourcing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 4, pp. 901–914, 2018. [Online]. Available: https://doi.org/10.1109/TPDS.2017.2775232

[26] J. Xu and Z. Rao and L. Xu and D. Yang and T. Li, "Incentive Mechanism for Multiple Cooperative Tasks with Compatible Users in Mobile Crowd Sensing via Online Communities," *IEEE Transactions on Mobile Computing*, vol. 19, no. 7, pp. 1618–1633, 2020.

[27] J. Liu and Y. Yang and D. Li and x. deng and S. Huang and H. Liu, "An Incentive Mechanism Based on Behavioural Economics in Location-based Crowdsensing Considering an Uneven Distribution of Participants," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[28] F. Yang, J. Lu, Y. Zhu, J. Peng, W. Shu, and M. Wu, "Heterogeneous task allocation in participatory sensing," in *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*, 2015, pp. 1–6. [Online]. Available: https://doi.org/10.1109/GLOCOM.2014.7417173

[29] Y. Liu, B. Guo, Y. Wang, W. Wu, Z. Yu, and D. Zhang, "Taskme: multi-task allocation in mobile crowd sensing," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016, Heidelberg, Germany, September 12-16, 2016*, 2016, pp. 403–414. [Online]. Available: https://doi.org/10.1145/2971648.2971709

[30] X. Miao, K. Liu, L. Chen, and Y. Liu, "Quality-aware online task assignment in mobile crowdsourcing," in *12th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2015, Dallas, TX, USA, October 19-22, 2015*, 2015, pp. 127–135. [Online]. Available: https://doi.org/10.1109/MASS.2015.40

[31] W. Gong, B. Zhang, and C. Li, "Location-based online task assignment and path planning for mobile crowdsensing," *IEEE Trans. Vehicular Technology*, vol. 68, no. 2, pp. 1772–1783, 2019. [Online]. Available: https://doi.org/10.1109/TVT.2018.2884318

[32] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, "Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning," *Discret. Optim.*, vol. 19, pp. 79–102, 2016. [Online]. Available: https://doi.org/10.1016/j.disopt.2016.01.005

[33] H. Zhao and M. Xiao and J. Wu and Y. Xu and H. Huang and S. Zhang, "Differentially Private Unknown Worker Recruitment for Mobile Crowdsensing Using Multi-Armed Bandits," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[34] B. Zhao and S. Tang and X. Liu and X. Zhang and W. Chen, "iTAM: Bilateral Privacy-Preserving Task Assignment for Mobile Crowdsensing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[35] F. Wu and S. Yang and Z. Zheng and S. Tang and G. Chen, "Fine Grained User Profiling for Personalized Task Matching in Mobile Crowdsensing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[36] Y. Wu, Y. Wang, W. Hu, and G. Cao, "Smartphoto: A resource-aware crowdsourcing approach for image sensing with smartphones," *IEEE Trans. Mob. Comput.*, vol. 15, no. 5, pp. 1249–1263, 2016. [Online]. Available: https://doi.org/10.1109/TMC.2015.2444379

[37] H. Zheng and J. Wu, "Online to offline business: Urban taxi dispatching with passenger-driver matching stability," in *37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017, Atlanta, GA, USA, June 5-8, 2017*, 2017, pp. 816–825. [Online]. Available: https://doi.org/10.1109/ICDCS.2017.14

[38] M. Kümmel, F. Busch, and D. Z. W. Wang, "Taxi dispatching and stable marriage," in *The 7th International Conference on Ambient Systems, Networks and Technologies (ANT 2016) / The 6th International Conference on Sustainable Energy Information Technology (SEIT-2016) / Affiliated Workshops, May 23-26, 2016, Madrid, Spain*, 2016, pp. 163–170. [Online]. Available: https://doi.org/10.1016/j.procs.2016.04.112

[39] R. Zhang, X. Cheng, and L. Yang, "Flexible energy management protocol for cooperative ev-to-ev charging," *IEEE Trans. Intelligent Transportation Systems*, vol. 20, no. 1, pp. 172–184, 2019. [Online]. Available: https://doi.org/10.1109/TITS.2018.2807184

[40] E. Bulut, M. C. Kisacikoglu, and K. Akkaya, "Spatio-temporal non-intrusive direct v2v charge sharing coordination," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 9385–9398, 2019.

[41] A. Ismaili, N. Hamada, Y. Zhang, T. Suzuki, and M. Yokoo, "Weighted matching markets with budget constraints," *J. Artif. Intell. Res.*, vol. 65, pp. 393–421, 2019.

[42] Y. Kawase and A. Iwasaki, "Approximately stable matchings with budget constraints," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 1113–1120.

[43] "Amazon mechanical turk." [Online]. Available: https://www.mturk.com/.

[44] D. Rawitz and A. Rosén, "Online budgeted maximum coverage," in *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, 2016, pp. 73:1–73:17. [Online]. Available: https://doi.org/10.4230/LIPIcs.ESA.2016.73

[45] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong, "On the levy-walk nature of human mobility," *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 630–643, 2011.

[46] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong, "CRAWDAD dataset ncsu/mobilitymodels (v. 2009-07-23)," Downloaded from https://crawdad.org/ncsu/mobilitymodels/20090723, Jul. 2009.

Fatih Yucel (M'17) received B.S. degree in Gazi University in Turkey in 2017. He is now pursuing his PhD degree in the Computer Science Department of Virginia Commonwealth University under the supervision of Dr. Eyuphan Bulut. He joined MoWiNG lab in Fall 2017. He is working on development of stable task assignment algorithms for mobile crowd sensing applications. He is a member of IEEE.

Murat Yuksel (SM'11) received the BS degree in computer engineering from Ege University, Izmir, Turkey, in 1996, and the MS and PhD degrees in computer science from RPI, in 1999 and 2002, respectively. He is a professor with the ECE Department, University of Central Florida (UCF), Orlando, Florida. Prior to UCF, he was with the CSE Department, University of Nevada, Reno (UNR), Reno, Nevada, as a faculty member until 2016. He worked as a software engineer with Pepperdata, Sunnyvale, California, and a visiting researcher with AT&T Labs and the Los Alamos National Lab. His research interests include networked, wireless, and computer systems with a recent focus on big-data networking, UAV networks, optical wireless, public safety communications, device-to-device protocols, economics of cyber-security and cybersharing, routing economics, network management, and network architectures. He has been on the editorial board of Computer Networks, and published more than 100 papers in peer-reviewed journals and conferences. He is a senior member of the IEEE, and a senior and life member of the ACM.

Eyuphan Bulut (SM'20) received the Ph.D. degree in computer science from Rensselaer Polytechnic Institute (RPI), Troy, NY, in 2011. He then worked as a senior engineer in Mobile Internet Technology Group (MITG) group of Cisco Systems in Richardson, TX for 4.5 years. He is now an Associate Professor with the Department of Computer Science, Virginia Commonwealth University (VCU), Richmond, VA. His research interests include mobile and wireless computing, network security and privacy, mobile social networks and crowd-sensing. Dr. Bulut has been serving as an Associate Editor in IEEE Access. He is a senior member of the IEEE and a member of ACM.