Bernd Simon, Sumedh Dongare, Tobias Mahn, Andrea Ortiz and Anja Klein, ”Delay- and Incentive-Aware Crowdsensing: A Stable Matching Approach for Coverage Maximization,” in *Proc. of the IEEE International Conference on Communications - (IEEE ICC 2022)*, May 2022.

Delay- and Incentive-Aware Crowdsensing: A Stable Matching Approach for Coverage Maximization

Bernd Simon, Sumedh Dongare, Tobias Mahn, Andrea Ortiz, Anja Klein
Communications Engineering Lab, TU Darmstadt, Germany, {b.simon, s.dongare, t.mahn, a.ortiz, a.klein}@nt.tu-darmstadt.de

***Abstract*—Mobile crowdsensing (MCS) is a novel approach to increase the coverage, lower the costs, and increase the accuracy of sensing data. Its main idea is to collect sensor data using mobile units (MUs). The sensing is controlled by a mobile crowdsensing platform (MCSP) through the assignment of delay-sensitive sensing tasks to the MUs. Although promising, research effort in MCS is still needed to find task assignment solutions that maximize the coverage while considering the cost incurred by the MCSPs, the preferences of the MUs and the limited communication resources available. Specifically, we identify two main challenges: (i) A task assignment problem which incorporates the MCSP's utility and the preferences of the MUs. (ii) An underlying communication resource allocation problem formulating the requirement of the timely transmission of sensing results given the limited communication resources. To address these challenges, we propose a novel two-stage matching algorithm. In the first stage, potential MU-task pairs are constructed considering the preferences of the MUs and the utility of the MCSP. In the second stage, the communication resource allocation is done based on potential MU-task pairs from the first stage. Through numerical simulations, we show that our proposed approach outperforms state-of-the-art methods in terms of the MCSP's utility, coverage and MU's satisfaction.**

I. Introduction

Mobile Crowdsensing (MCS) refers to the collection of sensor data by a group, or a "crowd", of mobile units (MUs) [1]. In recent years, MCS has increasingly gained the attention of the research community and the industry due to the advantages it brings over traditional wireless sensor networks, i.e., larger coverage, lower costs and higher accuracy in the sensed data [2]. Thanks to the heterogeneity of the available mobile devices, MCS allows the collection of a diverse set of sensor data such as, measurements of environmental parameters like temperature, humidity, air quality index; traffic reports, or personal health-related data. As a consequence, it can be used in many different application scenarios like traffic monitoring and dynamic re-routing of the traffic (Waze, Foursquare), environmental monitoring systems [3], landmark reconnaissance and identification [4] and health-monitoring using mobile health applications [5].

In a typical MCS system, the interaction between a data requester and the MUs is done through a so-called Mobile Crowdsensing Platform (MCSP). In practice, the data request is treated by the MCSP as a sensing task which needs to be performed by a set of MUs. The MCSP uses a task assignment policy to assign these sensing tasks to suitable MUs. Selecting suitable MUs requires the consideration of available communication resources and monetary costs for the MCSP. A resource allocation is required for the limited communication resources, e.g. bandwidth, enabling a timely transmission of the sensed data. Monetary costs are caused by the necessity of providing incentives to the MUs to participate in the sensing task. As a result, the task assignment policy must solve the challenging coverage maximization problem while considering limited communication resources and monetary costs from the MCSP's perspective. Moreover, the MU's preferences for task execution should also be considered, as they play a crucial role in the user participation and satisfaction.

Research efforts have been put to overcome these challenges separately. In [3], the authors aim at maximizing the coverage under a budget constraint. They show the problem is NP-hard and proposed heuristic solutions that favor the allocation of task according to their deadline or location. A similar problem is considered in [4], where the authors consider the coverage quality provided by each user in the selection decision. The problem of energy minimization given a coverage constraint is investigated in [6]. To this aim, a framework that saves communication resources by combining the uploading of sensor data with the user's phone calls is proposed. In [7], the minimization of the user selection cost under a coverage constraint is studied. For this purpose, the authors consider a grid map to represent the target sensing area and assume that the user trajectories are known beforehand. In these works, the goal is to maximize the coverage under a budget constraint at the MCSP. However, the MU's preferences are not included. In [8], the authors consider a MCS budgeted coverage problem in which the conflicting monetary interests of the service requesters and MUs are studied. A bilateral preference-based stable matching solution is proposed in order to satisfy both entities. Although in [8] the interests of both, the requesters and the MUs, are considered, the authors do not take into account the communication cost associated to the sensing.

In this paper, we investigate the MCS coverage maximization problem as an interdependent joint task assignment and communication resource allocation problem. The considered sensing tasks have task-specific sizes and deadlines. We include the MCSP's and MU's preferences and formulate a two-staged matching problem to maximize the coverage. In the considered reference schemes, authors focus on a spectrum

This work has been funded by the German Research Foundation (DFG) as a part of the projects B3, C1 and T2 within the Collaborative Research Center (CRC) 1053 - MAKI (Nr. 210487104) and has been supported by the BMBF project Open6GHub (Nr. 16KISK014) and the Loewe Center EmergenCity.

sensing scenario and propose heuristic approaches for task assignment such that coverage will be maximized. Specifically, in [9] a greedy algorithm is proposed in which the users are selected based on their marginal contributions per cost value. In [10], the authors use a genetic algorithm to solve the task assignment problem. In contrast to these schemes, our proposed algorithm maximizes the coverage, solving the task assignment and resource allocation problem for deadline sensitive tasks. The proposed two-stage matching algorithm aims to find stable pairs of tasks and MUs that maximize the MCSP utility, i.e. coverage, as well as the MUs utility by considering their preferences. We consider a realistic scenario in which communication resources at the access points of the network are limited. The results of the numerical simulations indicate that our proposed algorithm based on two-stage matching outperforms these reference schemes.

The rest of the paper is structured as follows. Section II introduces the considered system model. In Section III, the MCP's and MU's perspectives are described and the coverage maximization problem is formulated. The proposed two-stage matching algorithm is explained in Section IV and the results of the numerical evaluation are presented in Section V. Finally, Section VI concludes the paper.

II. System model

This work considers the joint problem of task assignment and wireless bandwidth allocation for coverage maximization in a multi-task crowdsensing scenario. Two relevant stakeholders are considered; the MUs and the MCSP. K MUs are located in the area to be sensed, each equipped with different sensors to complete sensing tasks. The set of all K MUs is denoted by $\mathcal{K} = \{1, \dots, K\}$. The MCSP offers N different sensing tasks. Each task n is assumed to require a different sensor installed at the MUs. Examples of possible tasks are sensing the temperature, monitoring the traffic density or discovering locally available network resources, e.g. licensed or unlicensed spectrum. A sensing task can be assigned to multiple MUs to measure at different locations in the considered area. Therefore, the area to be sensed is divided into two-dimensional grids G_n, one grid for each task n, with cells $g_n(i,j) \in G_n$ with the horizontal and vertical indices $i, j \in \mathbb{N}$. To increase the location privacy of the participants, all MUs only report their current location cell $g_n(i,j)$ of grid G_n to the MCSP and not the exact location in the network. The MCSP aims to maximize the coverage area of each sensing task while satisfying a maximum time constraint for the arrival of the results. For the transmission of the sensing results from the MUs to the MCSP, M APs are located in the area with a total bandwidth B_m available at each AP. An overview over the system model is given in Figure 1.

The set of all sensing tasks is denoted by $\mathcal{N}$. Each MU k has a set of sensors installed, which enables it to participate in a subset $\mathcal{N}_k \subset \mathcal{N}$ of tasks. Each MU k can perform one sensing task from the set $\mathcal{N}_k$. A preference of the MU k for a sensing task n is defined by a preference relation $\succeq_k^{\mathrm{MU}}$, i.e. if $n' \succeq_k^{\mathrm{MU}} n$ the MU k prefers task n' over

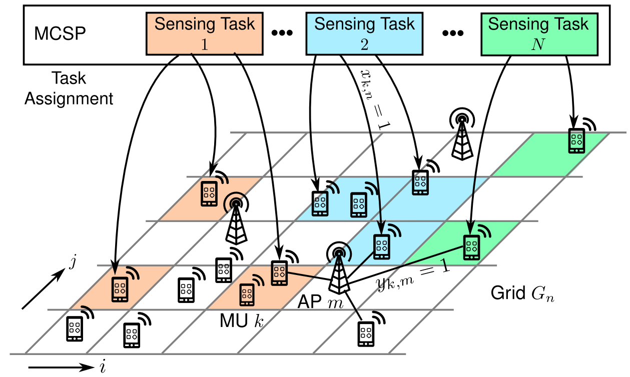

Fig. 1. Overview of the system model.

task n. A list of all tasks $n \in \mathcal{N}_k$ ordered according to the preference relation is called the *preference list* PL_k of MU k. The preference list PL_k can be influenced by the MCSP by using suitable incentive mechanisms. We consider two incentive mechanisms: (i) Micropayments [11], where the MCSP transfers $c_{k,n}$ monetary units, e.g. virtual currency, to MU k as a reward for the completion of sensing task n. (ii) The MU itself is a consumer of the crowdsensed data as proposed in [12]. The MCSP transmits a relevant selection of the sensing results in grid G_n to the MU as reward for the participation in task n. In this case, we assume $c_{k,n} = 0$.

The participation of MU k in task n is stored in a binary variable $x_{k,n}$. If MU k participates in task n, $x_{k,n} = 1$. Otherwise, $x_{k,n} = 0$. The sensing task allocation matrix $\{x_{k,n}\}$ of all K MUs and N tasks is denoted by $\mathbf{X}$. If MU k participates in task n, it covers an area of at least one grid cell $g_n(i,j)$. Depending on the sensing task n, a MU might also cover multiple neighboring cells. A binary coverage variable $\delta_{k,n}(i,j)$ is introduced to store the cells $g_n(i,j)$ covered in grid G_n by MU k. $\delta_{k,n}(i,j) = 1$ when the sensor of MU k covers cell $g_n(i,j)$ and $\delta_{k,n}(i,j) = 0$ if not. The coverage of a cell $g_n(i,j)$ of sensing task n by at least one MU is

$$\Delta_n(\mathbf{X}, i, j) = \begin{cases} 1, & \text{if } \sum_{k=0}^{K} x_{k,n} \delta_{k,n}(i,j) > 0 \\ 0, & \text{else.} \end{cases} \tag{1}$$

The number of all cells that are covered for sensing task n can be written as total coverage

$$C_n(\mathbf{X}) = \sum_{g_n(i,j) \in G_n} \Delta_n(\mathbf{X}, i, j). \tag{2}$$

As not all the cells of the network have the same importance to the MCSP [9], [10], we introduce a task and cell-specific utility $\alpha_{n,i,j}$ for each cell $g_n(i,j)$. This utility $\alpha_{n,i,j}$ has the same unit as the payments $c_{k,n}$ to the MUs, i.e. monetary units. The weighted coverage is given as

$$W_n(\mathbf{X}) = \sum_{g_n(i,j) \in G_n} \alpha_{n,i,j} \cdot \Delta_n(\mathbf{X}, i, j). \tag{3}$$

In addition to the sensing task assignment, a communication resource allocation has to be found. The bandwidth B_m of each AP m is divided into L_m^{AP} resource blocks with bandwidth

$b_m = \frac{B_m}{L_m^{\text{AP}}}$. Each resource block can be assigned to only one MU. We introduce the variable $y_{k,m}$ which is set to 1 if MU k uses AP m and set to 0 otherwise. The AP assignment matrix $\{y_{k,n}\}$ is denoted by $\mathbf{Y}$. If MU k participates in sensing task n, i.e. $x_{k,n} = 1$, the result with size s_n, measured in bits, needs to be transmitted from the MU k via AP m to the MCSP. The channel gain between MU k and AP m is denoted by $|h_{k,m}|^2$. The number of bits transmitted from MU k to AP m is

$$\Gamma_{k,m} = \log_2\left(1 + \frac{p^{\text{trans}}|h_{k,m}|^2}{\sigma^2}\right), \tag{4}$$

where σ^2 is the white Gaussian noise power. The time required for transmitting the result of sensing task n to AP m is

$$\tau_{k,m,n}^{\text{tx}} = \frac{s_n}{l_{k,m,n} \cdot b_m \cdot \Gamma_{k,m}}, \tag{5}$$

where $l_{k,m,n}$ denotes the number of resource blocks assigned to MU k at AP m. To fulfill the time constraint $\tau_n^{\max}$ for reporting the result to the MCSP, each MU requires

$$l_{k,m,n} = \left\lceil \frac{s_n}{b_m \cdot \Gamma_{k,m} \cdot \tau_n^{\max}} \right\rceil \tag{6}$$

communication resource blocks from the AP m.

The cost for the crowdsensing is the sum of all incentive payments $c_{k,n}$ to the MUs. The total utility for the MCSP of sensing task n for a task allocation $\mathbf{X}$ is given as

$$U_n^{\text{MCSP}}(\mathbf{X}) = W_n(\mathbf{X}) - \sum_{k=0}^{K} x_{k,n}c_{k,n}. \tag{7}$$

III. Problem Formulation

In this section, we formulate the task assignment and resource allocation problem considering the perspectives of the MUs and the MCSP. To establish a beneficial situation for all stakeholders, a stability condition is formulated. This stability condition ensures that neither the MCSP nor the MUs can improve by one-sided deviation from the stable outcome.

A. Different perspectives of the MCSP and the MUs

The MCSP aims to maximize the coverage of the crowdsensing. This problem can be written as

$$\max_{\mathbf{X},\mathbf{Y}} \quad \sum_{n=0}^{N}\left(W_n(\mathbf{X}) - \sum_{k=0}^{K} x_{k,n}c_{k,n}\right). \tag{8}$$

The objective function (8) is the sum of the utilities of all sensing tasks n. Furthermore, the formulation in (7) can be used to introduce a preference list for different task assignments $\mathbf{X}, \mathbf{X}'$. The MCSP prefers an allocation $\mathbf{X}$ of MUs over $\mathbf{X}'$ if the utility (7) is higher for $\mathbf{X}$ than for $\mathbf{X}'$. Formally, $\mathbf{X} \succeq_n^{\text{MCSP}} \mathbf{X}' \iff U_n^{\text{MCSP}}(\mathbf{X}) \geq U_n^{\text{MCSP}}(\mathbf{X}')$.

The sensing task selection of the MU depends on its individual preferences. As introduced in Section II, each MU has preferences $\succeq_k^{\text{MU}}$ ranking all sensing tasks n. The preference list PL_k of each MU is constructed as follows: the most preferred task is a task for which the MU requires the data. In this case the MU participates without a micropayment, i.e. $c_{k,n} = 0$. The remaining tasks are ranked according to the micropayments $c_{k,n}$ paid by the MCSP. The objective of each MU is to get a task which is on top of its preference list.

B. Formulation as matching problem

To incorporate the preferences of the MCSP and the MUs, the problem is formulated as a stable matching problem. In the following, the sensing task that is assigned to MU k is denoted by $\mu_{\text{Task}}^{\text{MU}}(k)$, formally $\mu_{\text{Task}}^{\text{MU}}(k) = \{n \in \mathcal{N} \,|\, x_{k,n} = 1\}$. Additionally, all MUs that are assigned to sensing task n are denoted by $\mu_{\text{Task}}^{\text{MCSP}}(n) = \{k \in \mathcal{K} \,|\, x_{k,n} = 1\}$.

We define stability as the absence of a blocking pair [13]:

Definition 1 (Blocking pair)**.** *A task assignment $\mathbf{X}$ is blocked by a set of MUs $C \subseteq \mathcal{K}$ and a sensing task $n \in \mathcal{N}$, if all of the following conditions hold:*

- (i) *$C \setminus \mu_{Task}^{MCSP}(n) \neq \varnothing$, meaning that not all MUs in C are already assigned to the sensing task n.*
- (ii) *$C \succ_n^{MCSP} \mu_{Task}^{MCSP}(n)$, meaning that sensing task n prefers the set C over its currently assigned MUs $\mu_{Task}^{MCSP}(n)$.*
- (iii) *$m \succ_k^{MU} \mu_{Task}^{MU}(k), \forall k \in C$, meaning that all MUs in C prefer the sensing task n over their current assigned sensing task $\mu_{Task}^{MU}(k)$.*

If a blocking pair exists, the MUs in C could change to their preferred task n and the MCSP's utility of task n would increase. Therefore a solution where a blocking pair exists is considered unstable. The stable matching problem is given by

$$\text{Find a stable task assignment } \mathbf{X} \tag{9}$$

$$\text{s.t.} \quad \sum_{k=0}^{K}\sum_{n=0}^{N} x_{k,m}y_{k,m}l_{k,m,n} \leq L_m^{\text{AP}} \quad \forall m \tag{9a}$$

$$\sum_{n=0}^{N} x_{k,n} \leq 1, \quad \sum_{m=0}^{M} y_{k,m} \leq 1 \quad \forall k \tag{9b}$$

$$\sum_{m=0}^{M} y_{k,m} = \sum_{n=0}^{N} x_{k,n} \quad \forall k \tag{9c}$$

$$x_{k,n}, y_{k,m} \in \{0,1\} \tag{9d}$$

Constraint (9a) considers the limited number of available communication resource blocks at each AP m. The fact that each MU can be matched to at most one task and one AP, is formulated in constraint (9b). Constraint (9c) includes the requirement that MU k is allocated to an AP if it participates in sensing task n.

Note that there are $2^{|\mathcal{K}|}$ solutions for C, and $|\mathcal{N}|$ combinations to chose n. An exhaustive search algorithm enumerating all blocking pairs and successively removing them from the matching has a complexity of $O(|\mathcal{N}| \cdot 2^{|\mathcal{K}|})$, which makes it infeasible for practical applications. For example, if $|\mathcal{K}| = 200$ and $|\mathcal{N}| = 4$, there are over $6.4 \cdot 10^{60}$ possible blocking pairs. Motivated by this high complexity, we propose a two-stage matching solution in the next section.

Algorithm 1 Proposed Two-Stage Matching

Require: Set of MUs $\mathcal{K}$, sensing tasks $\mathcal{N}$ and APs $\mathcal{M}$.

1: Create initial preference lists $\mathrm{PL}_k^{(0)}$, $i \leftarrow 0$.
2: **repeat**
3: *% Stage 1: Stable Task Assignment (STA)*
4: $\mu_{\text{Task}}^{(i)} \leftarrow \mathrm{STA}(\mathcal{K}, \mathcal{N}, \mathrm{PL}_k^{(i)})$ ▷ Alg. 2
5: $\mathcal{K}_{\text{Task}}^{(i)} \leftarrow$ Set of MUs that are matched to a task in $\mu_{\text{Task}}^{(i)}$.
6: *% Stage 2: Communication Resource Assignment (CRA) for $\mathcal{K}_{Task}$*
7: $\mu_{\text{AP}}^{(i)} \leftarrow \mathrm{CRA}(\mathcal{K}_{\text{Task}}, \mathcal{M}, w)$ ▷ Alg. 3
8: $\mathcal{K}_{\text{AP}}^{(i)} \leftarrow$ Set of MUs that are matched to an AP in $\mu_{\text{AP}}^{(i)}$.
9: $\mathrm{PL}_k^{(i+1)} = \mathrm{PL}_k^{(i)}$ ▷ Update the preferences
10: **for all** $k \in \mathcal{K}_{\text{Task}} \setminus \mathcal{K}_{\text{AP}}$ **do**
11: $\mathrm{PL}_k^{(i+1)} \leftarrow \mathrm{PL}_k^{(i)} \setminus \mu_{\text{Task}}^{(i)}(k)$
12: **end for**
13: $i \leftarrow i+1$
14: **until** $\mathcal{K}_{\text{Task}}^{(i)} = \mathcal{K}_{\text{AP}}^{(i)}$ ▷ Constraint (9c)
15: **return** $\mu_{\text{Task}}^{(i)}, \mu_{\text{AP}}^{(i)}$

Algorithm 2 Stable Task Assignment $\mathrm{STA}(\mathcal{K}, \mathcal{N}, PL_k)$

Require: Set of MUs $\mathcal{K}$, sensing tasks $\mathcal{N}$ and preference lists PL_k

1: **for** $i = 0, \dots, |\mathcal{N}|$ **do**
2: Every unmatched MU k proposes to the task n^* on top of PL_k and is added to the set of proposing users σ_{n^*}.
3: **for** $j = 0, \dots, |\mathcal{N}|$ **do**
4: **while** $\sigma_j \neq \{\}$ **do**
5: $k^* = \max_{k \in \sigma_j} U_n^{\text{MCSP}}(\mu_{\text{Task}}^{\text{MCSP}}(j) \cup k) - U_n^{\text{MCSP}}(\mu_{\text{Task}}^{\text{MCSP}}(j))$
6: $w_{k^*} = U_n^{\text{MCSP}}(\mu_{\text{Task}}^{\text{MCSP}}(j) \cup k^*) - U_n^{\text{MCSP}}(\mu_{\text{Task}}^{\text{MCSP}}(j))$
7: **if** $w_{k^*} > 0$ **then**
8: $\mu_{\text{Task}}^{\text{MCSP}}(j) \leftarrow \mu_{\text{Task}}^{\text{MCSP}}(j) \cup k^*$
9: **end if**
10: $PL_{k^*} \leftarrow PL_{k^*} \setminus j$ ▷ Remove preference
11: $\sigma_j \leftarrow \sigma_j \setminus k^*$ ▷ Remove k^* from the proposing MUs
12: **end while**
13: **end for**
14: **end for**
15: **return** $\mu_{\text{Task}}, \{w_{k,n}\}$ ▷ Task assignment & marginal contributions

Algorithm 3 Comm. Resource Allocation $\mathrm{CRA}(\mathcal{K}, \mathcal{M}, w)$

Require: Set of MUs $\mathcal{K}$, APs $\mathcal{M}$ and marginal contribution w

1: Construct a list $\mathrm{PL}_k^{\text{AP}}$ for each MU for the APs according to $|h_{k,m}|$
2: **for** $i = 0, \dots, |\mathcal{M}|$ **do**
3: Every unmatched MU k proposes to the AP m^* on top of $\mathrm{PL}_k^{\text{AP}}$ and is added to the set of proposing users σ_m.
4: **for** $j = 0, \dots, |\mathcal{M}|$ **do**
5: $\mathrm{PL}_{k^*}^{\text{AP}} \leftarrow \mathrm{PL}_{k^*}^{\text{AP}} \setminus j$
6: $\mu_{\text{AP}}(j) \leftarrow \mathrm{KNAPSACK}(\sigma_j, \mathrm{CAPACITY} = L_m^{\text{AP}}, \dots$ $\mathrm{WEIGHTS} = l_{k,m,n}, \mathrm{VALUES} = w_k)$
7: **end for**
8: **end for**
9: **return** μ_{AP} ▷ Communication resource allocation

IV. Two-Stage Matching Algorithm

A. Two-stage matching

Our proposed algorithm consists of two matching stages: The stable task assignment and the communication resource allocation. As both problems are interdependent, they cannot be solved independently. The intuition for the proposed algorithm is the following. In the first stage the task assignment $\mathbf{X}$ is calculated considering the MU's and the MCSP's preferences under the assumption of unlimited communication resources. This yields the best-case task assignment, which is infeasible because of the communication constraint (9a). The next stage solves the association between MUs, which have a task assigned from stage 1, and APs. MUs which are found to be suboptimal in stage 2 are removed, and the assigned task is removed from the respective preference list. In the next iteration the matching is performed with the updated preferences. This algorithm is performed iteratively until each MU which is assigned to a task in stage 1 also is allocated to communication resources in stage 2.

The two-stage matching procedure is formalized in Algorithm 1. The algorithm requires the set of MUs $\mathcal{K}$, the tasks $\mathcal{N}$ and the APs $\mathcal{M}$. For each MU the preference for each task is initialized (line 1). The set of MUs and tasks is then given to the stable task assignment, described in the next section.

B. Stable Task Assignment

The Stable Task Assignment (STA) is presented in Algorithm 2. The procedure is based on a generalized many-to-one deferred-acceptance algorithm [13]. Each MU k proposes to participate in the most preferred task on their preference list PL_k (line 2). The set of all the MUs proposing to participate in task n is denoted by σ_n. Firstly, for each task, the marginal contribution to the MCSP's utility (8) of all proposing MUs in σ_n is calculated as

$$w_{k,n} = U_n^{\text{MCSP}}(\mu_{\text{Task}}^{\text{MCSP}}(n) \cup k) - U_n^{\text{MCSP}}(\mu_{\text{Task}}^{\text{MCSP}}(n)) \qquad (10)$$

The MU with the highest $w_{k,n}$ is selected (line 5), and is assigned to the sensing task if $w_{k,n}$ is positive (line 7-9). Next, this MU is removed from σ_n (line 10 and 11). This procedure is repeated until all proposing MUs are either accepted or deferred. After all MUs are either accepted by a sensing task or deferred by each task, the stable task allocation is finished. The next stage is that all MUs in $\mathcal{K}_{\text{Task}}$ with a sensing task assigned are passed to the Communication Resource Assignment algorithm, which is described in the next section.

C. Communication Resource Allocation

The Communication Resource Allocation (CRA) is shown in Algorithm 3. The input of this algorithm is the set $\mathcal{K}$ of MUs, the set $\mathcal{N}$ of APs and the marginal contributions $w_{k,n}$ of each MU to the assigned task. In line 1 each MU creates a preference list containing each AP, which is ordered according to the absolute value of the channel coefficient $|h_{k,m}|$. Then each MU proposes to connect to the AP on top of its preference list. After each AP received all proposals, the AP decides which MUs are accepted. To approximate the optimal allocation of the L_m^{AP} resource blocks, we model the allocation problem as a knapsack problem [14]. The capacity of the knapsack is given by the L_m^{AP} resource blocks. Each MU requires $l_{k,m,n}$ resource blocks (6) and contributes with $w_{k,n}$ to the utility of the MCSP. Algorithm 1 terminates when all MUs that are matched to one task are also assigned to an AP.

D. Stability of the task allocation

In the following, we show that the output of the proposed algorithm is stable according to Definition 1. The following cases may occur for each MU $k \in \mathcal{K}$ after Algorithm 1: (1) MU k is assigned to task n at the top of PL_k. As it is already the best outcome for MU k, no other sensing task leads to an improvement. (2) MU k is assigned to a task n' not at the top of its PL_k. This means, k has proposed in some iteration $i < I$ to participate in sensing task n in stage 1 and was rejected. If k was rejected by n then the marginal contribution $w_{k,n}$ to task n is negative. In this case the MU prefers task n over its current task n', but the MCSP's utility would decrease if MU k changes the task. (3) MU k is unmatched because it was removed in stage 1 because of one of the following reasons: Either MU k prefers not to contribute to the crowdsensing, and therefore never proposed to contribute to a task, or the marginal contribution $w_{k,n}$ of MU k to all sensing tasks is negative, e.g. due to a high cost $c_{k,n}$ to incentivize the MU, and therefore it would be irrational for the MCSP to accept the proposal of MU k. (4) MU k is unmatched because it was removed in stage 2: In this case, both the MCSP and the MU k agreed on a task allocation in stage 1, but no communication resources are available to transmit the sensing results before the deadline $\tau_n^{\max}$.

In all the presented cases it is not possible for both stakeholders, the MCSP and the MU, simultaneously to improve the task allocation to a more preferred one. Therefore, the proposed algorithm is said to have a stable outcome.

V. Numerical evaluation

A. Simulation setup

In this section, numerical results for the evaluation of the proposed two-stage matching algorithm are presented. The MCSP has $N = 4$ different sensing tasks with different result sizes. Two tasks of small size $s_n = 300\,\text{kbit}$ and two tasks of large size $s_n = 600\,\text{kbit}$ are considered. The deadline for the arrival of the sensing data is $\tau_n^{\max} = 100\,\text{ms}$. The sensors of MU k are assumed to cover all cells within a $r_n = 100\,\text{m}$ radius, i.e. $\delta_{k,n}(i,j) = 1$ for all cells within this circle. The MU's locations are generated by a homogeneous Poisson Point Process. The MCSP has a utility coefficient $\alpha_{n,i,j}$ for task n in cell $g_n(i,j)$ which is uniformly distributed in the interval $[0,5]$ in each cell. We assume that $10\,\%$ of the MUs contribute to a sensing task because they want to retrieve the sensing results from the MCSP, i.e., $c_{k,n} = 0$. The remaining MUs only contribute in exchange of a micropayment $c_{k,n}$, determined by the MCSP. Table I summarizes the considered parameters.

As reference, two state-of-the-art algorithms from the literature are considered: *Cell-wise greedy algorithm:* comparable to the approach in [9], for each MU and sensing task pair (k,n) in a cell, the marginal contribution $w_{k,n}$ (10) is calculated. The MU-task pair with the highest non-negative $w_{k,n}$ is selected. The sensing result is transmitted to the AP which minimizes the resource demand $l_{k,m,n}$ (6). Afterwards, the algorithm calculates the marginal contributions for the remaining MUs in the cell and selects the next MU greedily. This procedure is repeated until all MUs have a task assigned in each cell, or the remaining MUs have a negative marginal contribution. *Centralized genetic algorithm*: as in [10], a genetic algorithm is used to find the task and AP allocation for the MUs. We define a chromosome as the combination of task allocation $\mathbf{X}$ and AP allocation $\mathbf{Y}$, where $x_{k,n}$ and $y_{k,m}$ are the genes in the chromosome. As usually done in genetic algorithms, the mutation, crossover and selection steps are iteratively performed to the whole population of chromosomes $\mathcal{P} = (\mathbf{X}, \mathbf{Y})$. The algorithm terminates after a fixed number of iterations and returns the chromosome $(\mathbf{X}, \mathbf{Y})$ with the highest utility of the MCSP (8). In this case, the mutation and the crossover probabilities are set to $\frac{4}{K}$, so that in each mutation and crossover phase an average of 4 MUs change the task or AP. We set $|\mathcal{P}| = 10$ and the number of iterations to $\frac{|\mathcal{K}|\cdot|\mathcal{M}|\cdot|\mathcal{N}|}{|\mathcal{P}|}$.

TABLE I
Evaluation parameters

Parameter	Value
Size of simulated grid G_n	$1\,\text{km}^2$
Size of each quadratic cell $g_n(i,j)$	$25\,\text{m} \cdot 25\,\text{m}$
Density of MUs	$[0,500]\,\text{km}^{-2}$
Density of APs	$6\,\text{km}^{-2}$
Total Bandwidth B_m per AP m	$10\,\text{MHz}$
Bandwidth b_m per resource block	$200\,\text{kHz}$
Noise power σ^2	$10^{-13}\,\text{W}$
Transmit power p_k of MU k	$200\,\text{mW}$
Channel gain $\lvert h_{k,m}\rvert^2$	$\sim d^{-3}$ (Urban scenario)
Number of sensing tasks N	4 (2 small, 2 large)
Size s_n of the sensing task result	Small: $300\,\text{kbit}$
	Large: $600\,\text{kbit}$
Deadline $\tau_n^{\max}$ of the sensing task result	$100\,\text{ms}$
Weight $\alpha_{n,i,j}$ for a cell $g_n(i,j)$	Uniform in $[0,5]$
Micropayment $c_{k,n}$ for MU k for task n	$\frac{1}{2}\sum_{i,j}\alpha_{n,i,j}\delta_{k,n}(i,j)$
Radius r_n of sensing area	$100\,\text{m}$

B. Numerical results

For the evaluation, we consider the MCSP's utility, the coverage achieved for the sensing tasks and the satisfaction of the MUs with the task assignment. The utility of the MCSP (8) for a varying number of MUs is shown in Figure 2. For $K < 200$ MUs, all algorithms achieve approximately the same utility for the MCSP, as the communication resources are sufficient to enable the transmission of all completed task results from the MUs to the MCSP. With an increasing number of MUs $K > 200$, the proposed approach results in a higher MCSP's utility compared to the greedy or the genetic algorithm. With limited communication resources it is beneficial, in some cases, to assign multiple smaller tasks with less utility per task to MUs instead of one large task. The proposed algorithm takes this into account by solving a knapsack problem in the communication resource allocation. The greedy algorithm, however, assigns tasks only based on the MCSP's utility. This is a near-optimal strategy when sufficient communication resources are available [9], but it will lead to a depletion of communication resources without considering smaller tasks.

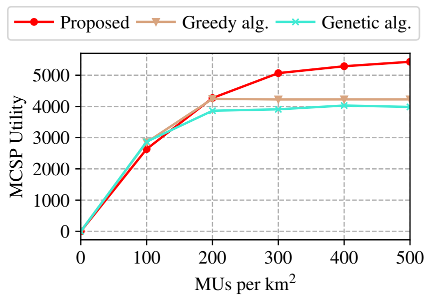

Fig. 2. Utility of the MCSP

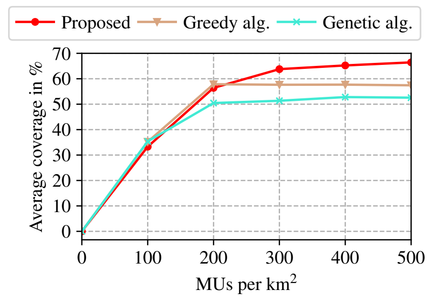

Fig. 3. Average coverage of sensing tasks

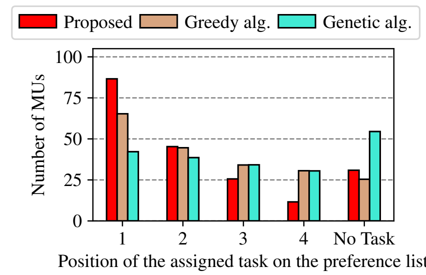

Fig. 4. Position of the assigned task on the preference list for $K = 200$ MUs

A larger number of MUs increases the difficulty of the task assignment and resource allocation problems. The probability of randomly selecting feasible solutions decreases when more MUs participate. This explains the decreasing performance of the genetic algorithm, as many mutations are not feasible solutions. The proposed algorithm avoids this pitfall by discarding infeasible MU-task pairs successively in stage 2, which reduces the number of potential MU-task pairs in each iteration. Tasks which could not be timely transmitted to the MCSP are removed from the respective preference list $\mathrm{PL}_k^{(i)}$ in each iteration i. Therefore, the knowledge of infeasible tasks is available at each MU. As MUs only propose to tasks on $\mathrm{PL}_k^{(i)}$, the communication overhead reduces with each iteration.

The average coverage of all sensing tasks is shown in Figure 3. There are two reasons why the coverage cannot be increased to 100%, even with a large number of MUs. The first reason is that the limited communication resources constrains the number of MUs contributing to each task. Furthermore, if the micropayment $c_{k,n}$ to the MU k is higher than its marginal contribution $w_{k,n}$ to sensing task n, then it would be irrational for the MCSP to accept MU k.

To evaluate the satisfaction of the MUs with the task assignment, the position on the preference list of the task assignment is shown in Figure 4. In the task assignment of the proposed algorithm 87 MUs are assigned to their most preferred task, whereas only 63 and 38 MUs get their most preferred task with the greedy and genetic algorithms, respectively. The proposed matching approach explicitly considers the MUs preferences for tasks in stage 1, including the incentives required for the MU, and therefore the satisfaction of the MUs with the task assignment is higher compared to the benchmark algorithms. The proposed algorithm has two unmatched MUs more than the greedy algorithm, since either the MUs have low preferences for the given sensing tasks or the MCSP would achieve a low utility from the MU's contribution.

VI. Conclusion

In this work, a coverage maximization problem for mobile crowdsensing with delay-sensitive sensing tasks is studied. Furthermore, two different incentives are considered: Micropayments by the MCSP or the sharing of the sensing results with the MUs. A stable task assignment problem incorporating the preferences of the MUs and a communication resource allocation problem is formulated. To solve the task assignment and communication resource allocation, a novel two-stage matching algorithm is proposed, and it is shown that the outcome of the proposed algorithm is stable. Through numerical simulations we show that the proposed method yields 27% more utility for the MCSP, 12% higher coverage of sensing and 40% more MUs get their first choice of tasks compared to state-of-the-art schemes.

References

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, 2011.
[2] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 29–35, 2014.
[3] H. To, L. Fan, L. Tran, and C. Shahabi, "Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints," in *IEEE Int. Conf. on Pervasive Comput. and Commun.*, 2016, pp. 1–8.
[4] 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. on Veh. Technol.*, vol. 65, no. 9, pp. 7698–7707, 2016.
[5] R. Pryss, J. Schobel, and M. Reichert, "Requirements for a flexible and generic API enabling mobile crowdsensing mhealth applications," in *Int. Workshop on Requirements Engineering for Self-Adaptive, Collaborative, and Cyber Physical Systems (RESACS)*, 2018, pp. 24–31.
[6] D. Zhang, H. Xiong, L. Wang, and G. Chen, "Crowdrecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint," in *Proc. ACM Int. Joint Conf. on Pervasive and Ubiquitous Comput.*, New York, 2014.
[7] N. Wang and J. Wu, "Cost-Efficient Heterogeneous Worker Recruitment under Coverage Requirement in Spatial Crowdsourcing," *IEEE Trans. on Big Data*, vol. 7, no. 2, pp. 407–420, 2021.
[8] F. Yucel, M. Yuksel, and E. Bulut, "Coverage-Aware Stable Task Assignment in Opportunistic Mobile Crowdsensing," *IEEE Trans. on Veh. Technol.*, vol. 70, no. 4, pp. 3831–3845, 2021.
[9] D.-H. Shin, S. He, and J. Zhang, "Joint sensing task and subband allocation for large-scale spectrum profiling," in *IEEE Conf. on Comput. Commun.*, Hong Kong, 2015, pp. 433–441.
[10] O. S. Mossad and M. ElNainay, "Genetic Algorithm-based Crowdsensing for Cognitive Radio Networks," in *Int. Conf. on Wireless and Mobile Comput., Networking and Commun.*, Limassol, 2018, pp. 1–7.
[11] S. Reddy, D. Estrin, M. Hansen, and M. Srivastava, "Examining Micro-Payments for Participatory Sensing Data Collections," in *Proc. ACM International Conference on Ubiquitous Computing*, New York, NY, USA, 2010, p. 33–36.
[12] M. Zhang, B. Swenson, J. Huang, and H. V. Poor, "Truthful Mobile Crowd Sensing with Interdependent Valuations," in *Proc. Int. Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, New York, 2020, p. 91–100.
[13] Y. Kawase, "Approximately Stable Matchings with General Constraints," in *Proc. of the 19th Int. Conf. on Autonomous Agents and MultiAgent Systems*. Int. Foundation for Autonomous Agents and Multiagent Systems, 2020, p. 602–610.
[14] V. V. Vazirani, *Approximation Algorithms*. Berlin, Heidelberg: Springer-Verlag, 2001.