IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Digital Object Identifier 10.1109/ACCESS.2018.DOI

Efficient Algorithms for Flexible Sweep Coverage in Crowdsensing

PEIHUANG HUANG[1], WENXING ZHU[2], KEWEN LIAO[3], TIMOS SELLIS[3], (Fellow, IEEE), ZHIYONG YU[2],LONGKUN GUO[2]
[1]College of Physics and Information Engineering, Fuzhou University, Fujian,China
[2]College of Mathematics and Computer Science, Fuzhou University, Fujian, China
[3]Swinburne University of Technology, Melbourne, Australia

Corresponding author: Longkun Guo (e-mail: lkguo@fzu.edu.cn).

This work is supported by Natural Science Foundation of China (Nos.61772005, 61672005, 61300025, 61772136) and Natural Science Foundation of Fujian Province (No.2017J01753).

ABSTRACT Sweep coverage is an important covering technique in mobile crowdsensing, in which users or participants are employed to periodically monitor a set of points of interest (POIs) each with a weight indicating the value of its information to be collected. Traditionally, each user proposes a route along which there are a set of POIs to be monitored. The task is to select a set of participants such that the total weight of the monitored POIs is maximized. However, in real applications, users should have the flexibility to offer several preferred routes. This arises our studied maximum Sweep Assignment problem with Flexibility (maxSAF), where each participant proposes several routes and the new task is to strategically assign each participant a route among her choices in which way maximizes the total weight of the monitored POIs. In this paper, we first prove the problem is $\mathcal{NP}$-complete and then devise two novel approximation algorithms with ratios 0.5 and 0.632. Experiments are also conducted to evaluate algorithms' practical performance. The results demonstrate that the proposed approximate methods are significantly faster (with up to two orders of magnitude runtime reduction) than the exact integer linear programming solution. In addition, we theoretically study another flexible sweep coverage model in which it costs to hire each user and the goal is to cover all POIs multiple times (for more complete and accurate information) while minimizing the total hiring cost.

INDEX TERMS Sweep assignment, $\mathcal{NP}$-complete, flexibility, crowdsensing, sensor networks.

I. INTRODUCTION

CROWDSENSING systems are attracting much interest from both academia and industry as a convenient way of collecting information or data of various qualities from points of interest (POIs) by assigning such tasks to hired participants (typically users with smart devices or sensors). Different from the traditional sensor networks which deploy limited sensors to continuously monitor POIs, crowdsensing systems rely on many moving and information-collecting agents. Typical such systems include the information collecting systems VTrack [27] and Nericell [21], environment noise collecting system NoiseTube [26], personal environmental impact report platform PEIR [22], recommendation framework SRMCS [28], and Query answering system CrowdK [16] which is also belong to crowdsourcing.

In traditional model of a crowdsensing system, each participant candidate first proposes one route. (for instance a trip between office and home). The system then finds the set of POIs covered by each route and performs participant selection. However, in practice such system should be more flexible in allowing participants to propose several preferred routes covering different sets of POIs. The benefits are two folds: i) from the system side, it increases the information quality (more quality POIs can be covered) or reduces the total hiring cost (less users can be employed to cover all POIs); ii) from the users side, the extra flexibility gives them more freedom and feasibility to participate. As a concrete example, a user may have several preferred routes between his office and home, consuming similar traffic time but are corresponding to totally different sets of POIs. With our introduced flexibility model, the system can benefit from better coverage by selecting from a broader range of routes, and on the other hand the users have more freedom with their choices. In this context, our goal is to assign each participant one of her proposed routes, such that the total number or weight of POIs monitored/covered by the assigned routes is maximized. This arises a new maximum Sweep Assignment problem with Flexibility (maxSAF) stated in the following:

Definition 1. Let $U = \{1, 2, \dots, n\}$ be a set of user ids and $P = \{1, \dots, m\}$ be a ground set of POI ids, where user $j \in U$ is associated with a collection of routes R_j and POI $i \in P$ is with a weight $w_i \in \mathbb{Z}^+$. Each route $l \in R_j$ is associated with or mapped to a set of POIs $S_l \subseteq P$ to be monitored by participant j periodically along the route l. The *maxSAF* problem aims to assign each $j \in U$ with a route $l_j \in R_j$, such that the total weight of their covered POIs, i.e.$\sum_{i \in S_U} w_i$ with $S_U = \bigcup_{j \in U} S_{l_j}$, is maximized.

For briefness, we assume there are q routes in total. Besides, we use $\mathcal{G}_j = \{S_l | l \in R_j\}$ to denote the collection of sets of POIs that can be monitored by user j along her proposed routes, and $\mathcal{S} = \cup_{j \in U} \mathcal{G}_j$ to denote the family of all possible route sets of POIs. To further guarantee the quality of collected information or data, POIs are required to be monitored by multiple participants. This arises a new minimum Multiple Sweep Coverage problem with Flexibility (minMSCF) as below:

Definition 2. Let $U = \{1, 2, \dots, n\}$ be a set of user ids and $P = \{1, \dots, m\}$ be a ground set of POI ids, where a user $j \in U$ is assigned with a hiring cost $c_j \in \mathbb{Z}^+$ and a collection of routes R_j. Each route $l \in R_j$ is associated with or mapped to a set of POIs $S_l \subseteq P$ to be monitored by participant j periodically along the route l. For a given integer $\delta > 0$, *minMSCF* aims to select a set of users $U' \subseteq U$ and assign each $j \in U$ with a route $l_j \in R_j$, such that $\sum_{j \in U'} c_j$ attains the minimum and each POI is covered by at least δ users, i.e. appears in at least δ sets of $\{S_{l_j} | l_j \in R_j, j \in U'\}$.

A. RELATED WORK

The coverage problem for crowdsensing systems is similar to the coverage problem in wireless sensor networks (WSN), such as: i) a sensor in a crowdsensing system corresponds to a participant with a mobile device; ii) both aim to monitor or cover POIs in a rigid way. In the following, we first show results on the coverage problem in wireless sensor networks, which has already attracted many interests from both academia and industry. Then we survey current results on POI coverage in crowdsensing systems, and at last present some theoretical results on the coverage problem itself.

The coverage problem can be divided into two categories: traditional continuous coverage and sweep coverage. The former studies how to use sensors to cover all POIs in an area (also known as full coverage) or a barrier [4], [17], [19], [30]. For the latter, the concept of sweep coverage was first developed from the context of robotics [3] and it requires periodically monitoring instead of continuously monitoring POIs. The goal of sweep coverage is to periodically monitor POIs with as few mobile sensors as possible. Cheng et al. [18] are the first to put forward a sweep coverage problem in WSN. They propose the Min Sensor Point Sweep Coverage problem (MSPSC) to find the minimum number of sensors to sweep cover POIs, which can be transformed into the travelling salesman problem and hence proven to be NP-hard. They further proved the problem cannot be approximated within a factor of 2 and gave a 3-approximate algorithm. Xi et al. [29] extends the MSPSC problem to be more practical – changing from considering static to dynamic POIs. Gorain et al. [8] provides a 2-approximation algorithm for solving sweep coverage for POIs and later devises a distributed approximation algorithm with the same factor. They have also introduced the problem of sweep coverage for an area of interest, proven its NP-completeness and proposed a $2\sqrt{2}$-approximation algorithm for a square area. Base on their previous work, Gorain et al. [7] study the problem of covering objects in the plane as linear segments and give a 2-approximation algorithm.

Sweep covering POIs via selecting participants is one of the core issues in crowdsensing. Reddy et al. [25] study the participant selection problem of crowdsensing for the first time. The goal is to select a predefined number of participants to maximize the spatial coverage. Cardone et al. [2] also propose a participant selection method based on the mobile crowdsensing platform. The difference is that in their model people who have more recently traversed the sensing task area are selected. They devise a greedy method to choose participants that maximize coverage utility while taking into consideration the previous coverage by existing selected participants. In [10], participants are selected based on their future positions, which are predicted with a human mobility model. In the work of [15], Krause et al. propose an algorithm to select a near-optimal subset of observations, using the demand weighted error reduction as a context-specific value of information. Liu et al. propose a novel participant selection method which considers energy-awareness for smartphone crowd sensing [20]. There are also some recent and relevant research works about participant selection in the crowdsensing paradigm [9], [31].

The above mentioned coverage problems has its root in theoretical computer science such as the maximum coverage (MC) problem and the minimum set cover (SC) problem, which are respectively maxSAF and minMSCF with $\delta = 1$ and when every user proposes exactly one route. It is known that MC admits an approximation ratio of $1 - \frac{1}{e}$ via a greedy algorithm [12]. The key idea is to repeatedly select a set with the maximum uncovered weight until all POIs are covered. The ratio is also known the best possible under the assumption $\mathcal{P} \neq \mathcal{NP}$, since MC admits no factor-$1 - \frac{1}{e} + \epsilon$ approximation provided $\mathcal{P} \neq \mathcal{NP}$ [6]. On the other hand, SC can be approximated within a factor of $1 + \ln |S|$ [13] but not approximable within $c \log n$ unless $\mathcal{P} = \mathcal{NP}$ for some $c > 0$ [6]. Further, SC remains $\mathcal{APX}-$complete and approximable within factor-$\sum_{i=1}^{k} \frac{1}{i} - 1/2$ [5] even when the cardinality of all sets is bounded by a given constant k. Moreover, even if the number of sets containing any element is also bounded by a constant $c \geq 2$, SC remains $\mathcal{APX}-$complete [24] and it is approximable within a factor c for both weighted and unweighted cases [1], [11].

B. OUR RESULTS AND TECHNIQUES

In this paper, we first show that the unweighted decision version of maxSAF is $\mathcal{NP}$-complete via a reduction from 3SAT, thus consequently it is also $\mathcal{NP}$-hard to solve the optimization version of maxSAF. We also prove some accompanying hardness results such as approximation lower bounds that maxSAF admits no approximation ratio better than $1 - \frac{1}{e}$ under the $\mathcal{P} \neq \mathcal{NP}$ assumption. For maxSAF, we first present an approximation algorithm with ratio $\frac{1}{2}$ and time complexity $O(mnq)$ based on a linear program (LP) primal-dual method [14], where m, n and q are respectively the numbers of POIs, users and routes. We further improve the ratio to $1 - \frac{1}{e}$ through a randomized LP rounding technique [14], which however requires solving LP and hence having a higher time complexity $O((m + n + q)^{3.5})$. As we will show later in Theorem 6, this ratio is already tight (the best possible) for the maxSAF problem. Besides theoretical analysis, we also empirically evaluate our algorithms primal-dual and randomized LP rounding for maxSAF through computer experiments. The results demonstrate their superior performance against the exact algorithm baseline of solving the problem ILP. Note that we compare our algorithm with ILP solution because maxSAF is practically a new problem for which there aren't other baselines to compare with except ILP.

In addition, with a dual-fitting technique [14] we present an $O(\ln m)$-approximation greedy algorithm for minMSCF for the special case that each participant proposes only one route. We have theoretically argued that its general case does not admit any non-trivial approximation.

C. ORGANIZATION

The remainder of this paper is organized as follows: Section II proves the necessary hardness results for problems maxSAF and minMSCF; Section III extensively studies the maxSAF problem with novel algorithms and their experiments; Section IV studies the minMSCF theoretically and presents a dual-fitting algorithm; and Section V concludes the paper.

II. COMPUTATIONAL COMPLEXITY OF *MAXSAF*

In this section, we will first prove the $\mathcal{NP}$-completeness of the decision maxSAF by giving a reduction from 3SAT and then show the inapproximability result of maxSAF via an L-reduct ion from the maximum coverage (MC) problem. Note that, we can not easily construct a reduction from MC to prove the $\mathcal{NP}$-completeness of the decision maxSAF, because MC model uses only k users while maxSAF can use as many users as needed.

A. THE $\mathcal{NP}$-COMPLETENESS OF MAXSAF

The decision maxSAF is: given a set of POIs $P = \{1, \ldots, m\}$, a set of users $U = \{1, 2, \ldots, n\}$ in which j is associated with a set of routes R_j, and a fixed integer K. The aim is to determine whether there exists an assignment that assigns each participant j with a route $r \in R_j$, such that the number of POI monitored by at least one participant is not less than K. Next we shall prove the following theorem:

Theorem 3. *The (unweighted) decision maxSAF is $\mathcal{NP}$-complete even when $K = |P|$.*

Hence, maxSAF is $\mathcal{NP}$-hard immediately from the above theorem.

The decision maxSAF is apparently in $\mathcal{NP}$, because any feasible solution to decision maxSAF is a polynomial size certificate and can be verified within polynomial time. It remains only to give the reduction from 3SAT to the decision unweighted maxSAF when $K = |P|$.

In an instance of 3SAT, we are given a set of variable X and a set of clauses $\mathcal{C}$ each of which is the OR of three literatures that is a variable of X or its negation. Then the corresponding instance of decision unweighted maxSAF is constructed as below:

1) For each clause $C_i \in \mathcal{C}$, add a POI i to P, which is initially an empty set; /*Construction of P.*/
2) For each variable $x_j \in X$, add a user j with two routes $R_j = \{1, 2\}$, which are associated with two sets $S_{j,1} = \{i | C_i \text{ contains } x_j\}$ and $S_{j,2} = \{i | C_i \text{ contains } \overline{x}_j\}$. Apparently, $S_{j,1}$ and $S_{j,2}$ are the sets of POIs corresponding to the clauses containing x_j and $\overline{x}_j$, respectively.

An example is depicted as in the following. For an instance of 3SAT:

$$C_1 = x_1 \vee \overline{x}_2 \vee x_4,\ C_2 = \overline{x}_1 \vee \overline{x}_2 \vee x_3,\ C_3 = x_2 \vee \overline{x}_3 \vee x_4,$$

$$C_4 = x_1 \vee \overline{x}_3 \vee \overline{x}_4,\ C_5 = x_1 \vee \overline{x}_2 \vee x_4,\ C_6 = x_2 \vee x_3 \vee x_4,$$

the corresponding instance of maxSAF is: $P = \{1, 2, 3, 4, 5, 6\}$ and $U = \{1, 2, 3, 4\}$, where R_j and its associated collection $\mathcal{G}_j$ of POI sets are as in the following:

$$\begin{aligned}
\mathcal{G}_1 &= \{S_{1,1} = \{1, 4, 5\}, S_{1,2} = \{2\}\}; \\
\mathcal{G}_2 &= \{S_{2,1} = \{3, 6\}, S_{2,2} = \{1, 2, 5\}\}; \\
\mathcal{G}_3 &= \{S_{3,1} = \{2, 6\}, S_{3,2} = \{3, 4\}\}; \\
\mathcal{G}_4 &= \{S_{4,1} = \{1, 3, 5, 6\}, S_{4,2} = \{4\}\}.
\end{aligned}$$

Then the correctness of Theorem 3 can be immediately obtained from the lemma below:

Lemma 4. *Any instance of 3SAT is satisfiable iff the constructed instance of decision maxSAF is feasible.*

Proof: Suppose an instance of 3SAT is feasible, then there exists a true assignment of the variables of X, say $\tau(x) \rightarrow \{true, false\}$ for $\forall x \in X$, such that each clause of $\mathcal{C}$ is true. A solution to the corresponding decision maxSAF instance can be constructed as follows: For each User j, if

$\tau(x_j) = true$, then assign $S_{l_j} := S_{j,1}$; Otherwise, assign $S_{l_j} := S_{j,2}$. Because τ satisfies all the clauses, the sets in $\cup_{j \in U} S_{l_j}$ will accordingly cover all the POIs. Besides, since x_j is either true or false (but not both), only one of the sets in $\{S_{j,1}, S_{j,2}\}$ can be selected. Hence, $\{S_{l_j} | j \in U\}$ is a feasible solution to the instance of decision maxSAF.

Conversely, suppose we have a feasible solution to maxSAF, say $\{S_{l_j} | j \in U\}$, in which the sets collectively cover all the POIs. Then the true assignment of the variables of X is as below: if $l_j = 1$, then set $\tau(x_j) = true$; Otherwise, set $\tau(x_j) = false$. Because l_j equals either 1 or 2, x_j will be assigned either true or false. Moreover, the assignment τ can satisfy all the clauses, because the sets of $\{S_{l_j} | j \in U\}$ collectively cover all the POIs. Therefore, τ is a feasible solution satisfying the instance of 3SAT. ■

Moreover, our reduction from 3SAT to maxSAF can be easily modified to obtain a reduction from 3-Occurrence 3SAT [14], a special case of 3SAT in which each variable appears at most 3 times. Then, because 3-Occurrence 3SAT is known $\mathcal{NP}$-complete, the unweighted decision maxSAF problem remains $\mathcal{NP}$-complete even for a very special case:

Corollary 5. *The unweighted decision* maxSAF *problem is $\mathcal{NP}$-complete even when $K = |P|$, each user has only two routes, each route associated with a set contains at most 3 POIs, and each POI appears in at most 3 route sets of the users.*

It is worth to note that, the decision form of maximum coverage, i.e. a special case of decision maxSAF when each user has only one route, is polynomial solvable. That is because it takes only linear time to simply verify whether the union of all the routes contain all the POIs in P.

B. INAPPROXIMABILITY

For maxSAF as an optimization problem we can show below that it is as hard as approximating MC, which admits no approximation algorithm with ratio better than $1 - \frac{1}{e}$ (the approximation lower bound) unless $\mathcal{P} \neq \mathcal{NP}$ [6]. Even worse for the general problem of minMSCF where each user proposes multiple routes, we also prove below that the problem does not admit any approximation, which is the reason why we only consider the special case when each user proposes only one route.

Theorem 6. *The maxSAF problem can not be approximated with a factor of $(1 - \frac{1}{e} + \epsilon)$ for any fixed real number $\epsilon > 0$, unless $\mathcal{P} = \mathcal{NP}$.*

Proof: We need only to give an L-reduction from MC. In any instance of the maximal coverage problem, we are given an integer k, $P = \{1, \ldots, m\}$ in which $i \in P$ has a nonnegative weight w_i, and a number of sets $S_1, \ldots, S_n$ that are subsets of P. The goal is to select k sets from $\{S, \ldots, S_n\}$, such that the number of POIs covered by the k sets is maximized. The construction of maxSAF is to simply construct a set of k users $U = \{1, \ldots, k\}$, where $j \in U$ is associated with $R_j = \{1, \ldots, n\}$, and the collection for R_j is $\mathcal{G}_j = \{S^j_1, \ldots, S^j_n\}$ and $S^j_i = S_i$. That is, $\mathcal{G}_1, \ldots, \mathcal{G}_k$ are actually k identical copies of $\{S, \ldots, S_n\}$.

Suppose we have a solution to maxSAF by selecting a set S_{l_j} for each $\mathcal{G}_j$, $\forall j$. Then, $\{S^1_{l_1}, \ldots, S^k_{l_k}\} = \{S_{l_1}, \ldots, S_{l_k}\}$, the collection of the k selected sets, is apparently a solution to MC with an identical weight-sum. Conversely, let a collection of k sets, say $\{S_1, \ldots, S_k\}$, be an solution of the instance of MC. Then $\{S^1_1, \ldots, S^k_k\}$ is immediately a solution to maxSAF with the same weight. This completes the proof. ■

Theorem 7. *The general* minMSCF *problem does not admit any non-trivial approximation, unless $\mathcal{P} = \mathcal{NP}$.*

Proof: From Theorem 3 we know that the decision maxSAF is $\mathcal{NP}$-complete when $K = |P|$. Then, suppose there is a non-trivial approximation algorithm for minMSCF when $\delta = 1$, then the algorithm is immediately a polynomial time algorithm for decision maxSAF when $K = |P|$, because decision maxSAF with $K = |P|$ is indeed minMSCF but without the minimization objective. This contradicts with the assumption $\mathcal{P} = \mathcal{NP}$. ■

III. ALGORITHMS FOR *MAXSAF*

In the section, we will first give an integer linear programming (ILP), together with its relaxation and the dual, for the the maximum Sweep Assignment problem with Flexibility (maxSAF). Then we present a primal-dual algorithm that can achieve an approximation ratio of 0.5, and an LP randomized rounding algorithm which can achieve an approximation ratio of $1 - \frac{1}{e}$.

A. AN ILP FORMULATION

We use $x_l \in \{0, 1\}$ to denote whether a route S_l is selected, and use $y_i \in \{0, 1\}$ to denote whether the i^{th} POI is covered. Then the integral linear programs for maxSAF is as below (ILP (1)):

$$
\begin{array}{llll}
\max & \sum_{i=1}^{m} w_i y_i & & \\
s.t. & y_i \leq \sum_{l:\, i \in S_l} x_l & \forall i \in P & (1)\\
 & \sum_{l:\, l \in R_j} x_l \leq 1 & \forall j \in U & (2)\\
 & x_l \in \{0,1\} & \forall l \in \cup_{j \in U} R_j & \\
 & y_i \in \{0,1\} & \forall i \in P &
\end{array}
$$

We can easily show *ILP (1)* correctly models maxSAF as below:

Proposition 8. *There exists a feasible solution to ILP (1) iff there exists a corresponding solution to maxSAF.*

Proof: Assume that there exists a feasible solution to maxSAF. Then we set $x_l = 1$ if route S_l is selected in the solution and $x_l = 0$ otherwise; Similarly, $y_i = 1$ if the i^{th} POI is covered and $y_i = 0$ otherwise. Hence, a solution to the ILP is obtained. It remains to show the solution is feasible against Inequality (1) and (2) within the ILP. Since the solution to *maxSAF* must satisfy the group constraint, Inequality (2) holds trivially. Moreover, the i^{th} POI, $\forall i$, is covered only if there exists a route which contains the POI is selected. That is, among the routes which contains the i^{th} POI, at least one of them must be selected. So we have $\sum_{l:\, i\in S_l} x_l \geq 1$. Then since $y_i \in \{0,1\}$, $\sum_{l:\, i\in S_l} x_l \geq y_i$ (i.e. Inequality (1)) must hold.

Conversely, assume that there exists a feasible solution to ILP (1). Then we select route S_l iff $x_l = 1$, and obtain a solution to maxSAF. It remains to show the feasibility of the solution. From Inequality (2), clearly the solution will satisfy the group constraint. In addition, for each POI i with $y_i = 1$, there must exist at least a route S_l with $x_l = 1$ and $i \in S_l$ following Inequality (1). That is, each POI i with $y_i = 1$ will be covered in the solution to maxSAF. This completes the proof. ■

B. A PRIMAL-DUAL ALGORITHM

We will first give a further reduced linear programming (LP) relaxation of ILP (1), since an immediately relaxation of ILP (1) can not have a promising dual. Note that $y_i \in \{0, 1\}$ can be relaxed to $y_i \leq 1$, as the aim of the formula is to maximize $\sum_{i=1}^{n} w_i y_i$ and $w_i \geq 0$ holds for $\forall i$. Consequently, $x_l \in \{0, 1\}$ can be relaxed to $x_l \geq 0$. So the LP relaxation of maxSAF is as below (LP (2)):

$$\begin{array}{lll}
\max & \sum_{i=1}^{m} w_i y_i & \\
s.t. & y_i - \sum_{l:\, i\in S_l} x_l \leq 0 & \forall i \in P \\
 & \sum_{l:\, l\in R_j} x_l \leq 1 & \forall j \in U \\
 & x_l \geq 0 & \forall l \in \cup_{j\in U} R_j \\
 & y_i \leq 1 & \forall i \in P
\end{array}$$

By the LP theory, we can immediately have the dual as follows (LP (3)):

$$\begin{array}{lll}
\min & \sum_{i=1}^{m} \beta_i + \sum_{j=1}^{n} \gamma_j & \\
s.t. & \sum_{j:\, l\in R_j} \gamma_j - \sum_{i\in S_l} \alpha_i \geq 0 & \forall l \in \cup_{j\in U} R_j \\
 & \alpha_i + \beta_i \geq w_i & \forall i \in P \\
 & \alpha_i, \beta_i, \gamma_j \geq 0 & \forall i \in P, \forall j \in U
\end{array}$$

For the relationship between the primal and dual as above, we have the following property:

Proposition 9. *For the primal and the dual LP, we have* $\min \sum_{i=1}^{m} \beta_i + \sum_{j=1}^{n} \gamma_j \geq \max \sum_{i=1}^{m} w_i y_i$.

The key idea of the primal-dual algorithm is to construct a feasible solution to the dual, and in the meantime, while also construct a feasible solution of maxSAF accordingly. The key of the construction is to guarantee that the value of the latter is not less than a bounded times of the former.

The main steps of constructing a dual solution proceeds as below: Firstly, set $\alpha_i = w_i$, $\beta_i = 0$ and $\gamma_j = \max_{l:\, l\in R_j} \left\{\sum_{i\in S_l} \alpha_i\right\}$ as their initial values; Secondly, find a promising S_l, and then increase the value of β_i for each $i \in S_l$ until $\beta_i = w_i$, while in the procession decrease α_i accordingly to keep $\alpha_i + \beta_i = w_i$. To find the promising S_l, our algorithm selects $\gamma_{j^*} = \max\{\gamma_j | j \in U\}$, and then $S_{l^*} \in \mathcal{G}_{j^*}$, such that $w(S_{l^*})$ attains maximum, i.e. $w(S_{l^*}) = \max\{w(S_l) | S_l \in \mathcal{G}_{j^*}\}$. The detailed algorithm is formally as in Algorithm 1.

Algorithm 1 A Primal-dual Algorithm for maxSAF.

Input: $U = \{1, \ldots, n\}$, $P = \{1, \ldots, m\}$, $\mathcal{G}_j = \{S_l | l \in R_j\}$ where $S_l \subseteq P$, $\mathscr{G} = \{\mathcal{G}_j | j \in U\}$ and a weight w_i for each $i \in P$;
Output: $\mathcal{S}' = \{S_{l_j} | j \in U\}$, a solution to *maxSAF*.

1: $\mathcal{S}' := \emptyset$;
2: **For** each POI i **do**
3: Set $\alpha_i := w_i$, $\beta_i := 0$;
4: **EndFor**
5: **For** each S_l **do**
6: Set $z_l := \sum_{i\in S_l} \alpha_i$;
7: **EndFor**
8: **For** each $\mathcal{G}_j$ **do**
9: Set $\gamma_j := \max_{l:\, S_l\in \mathcal{G}_j} \{z_l\}$;
10: **EndFor**
11: Find $\mathcal{G}_{j^*}$ such that $\gamma_{j^*} = \max\{\gamma_j\}$;
12: Find l^* such that S_{l^*} is the set with maximum weight $w(S_{l^*})$ in $\mathcal{G}_{j^*}$, and set $l_j := l^*$;
13: **For** each POI $i \in S_{l^*}$ **do**
14: Set $\beta_i := w_i$ and $\alpha_i = 0$;
15: For each S_l that contains i, set $z_l := z_l - \beta_i$ and $S_l := S_l \setminus \{i\}$;
16: **EndFor**
17: Set $\mathscr{G} := \mathscr{G} \setminus \{\mathcal{G}_{j^*}\}$;
18: **If** $\mathscr{G} \neq \emptyset$ **then** Go to Step 8;
19: **Else** Return $\mathcal{S}' = \{S_{l_j} | j \in U\}$.

Lemma 10. *The time complexity of Algorithm 1 is $O(mnq + n^2)$, where m, n and q are respectively the numbers of POIs, users and routes.*

Proof: Clearly, Step 8 to Step 18 as the out-loop will be repeated for $O(n)$ times, in which Step 11 and Step 12 respectively take $O(n)$ and $O(q)$ time, the for-loop starting at Step 13 takes $O(qm)$ time. Other steps takes relatively trivial time, so $O(mnq + n^2)$ time in total. ■

Theorem 11. *The approximation ratio of Algorithm 1 is $\frac{1}{2}$.*

Proof: According to the algorithm, $\beta_i = w_i$ if there exists a set $S \supseteq \{i\}$ that is selected and added in to $\mathcal{S}'$ in Step 14 and 15, and $\beta_i = 0$ otherwise. That is

$$\sum_{i=1}^{m} \beta_i \leq \sum_{i\in S\in \mathcal{S}'} w_i. \quad (3)$$

Assume that $\hat{S}_{l^*}$ is the set of POIs in S_{l^*} when it is selected for $\mathcal{G}_{j^*}$. Then clearly, initially in the iteration we have $\sum_{i\in \hat{S}_{l^*}} \beta_i = z_{l^*} = \gamma_{j^*}$. Afterwards as β_i is non-decreasing, and γ_l is non-increasing, then $\sum_{i\in \hat{S}_{l^*}} \beta_i \geq \gamma_{j^*}$ holds when the algorithm terminates. Moreover, let S be another set that is selected for $\mathcal{G} \in \mathscr{G}$, $\mathcal{G} \neq \mathcal{G}_{l^*}$, then we have $\hat{S}_{j^*} \cap \hat{S} = \emptyset$. Therefore, we have

$$\sum_{i=1}^{m} \beta_i = \sum_{\mathcal{G}_j \in \mathscr{G}} \sum_{\substack{i \in \hat{S} \\ \hat{S}\ selected\ for\ \mathcal{G}_j}} \beta_i \geq \sum_{l:\, \mathcal{G}_j \in \mathscr{G}} \gamma_j. \quad (4)$$

Combining Inequality (3) and (4) immediately yields a lower bound for the weight of the output of Algorithm 1:

$$\sum_{i \in S \in \mathcal{S}'} w_i \geq \frac{1}{2} \left(\sum_{i=1}^{m} \beta_i + \sum_{l:\, \mathcal{G}_l \in \mathscr{G}} \gamma_l \right).$$

Then combining with Proposition 9, we have $\sum_{i \in S \in \mathcal{S}'} w_i \geq \frac{1}{2} w^*_{LP} \geq \frac{1}{2} w_{OPT}$, where w^*_{LP} is the weight of an optimal solution to the primal LP (2) and w_{OPT} is the weight of an optimal solution to maxSAF. This completes the proof. ■

We can show the ratio analysis is already tight for Algorithm 1. An instance in which the algorithm can only output a solution with half of the total weight of an optimal solution is as follows. Let $P = \{1, \ldots, 2n\}$ be the set of POIs with uniform weight, and $U = \{1, \ldots, n, n+1, \ldots, 2n\}$ be the set of users where each $j \in \{1, \ldots, n\} \subset U$ is associated with two routes j_1 and j_2, whose POI sets are respectively $S_{j_1} = \{j\}$ and $S_{j_2} = \{n+j\}$ that each containing exactly one POI; while each $j \in \{n+1, \ldots, 2n\} \subset U$ is associated with exactly one route, say j_1, whose POI set $S_{j_1} = \{n+j\}$ contains only one POI. Then Algorithm 1 would select $\{n+j\}$ for each user $j \in \{1, \ldots, n\} \subset U$ in worst case, resulting in a total weight n; while the optimum solution, which assigns $\{j\}$ to each user $j \in U$, is with a total weight $2n$.

Corollary 12. *The approximation ratio $\frac{1}{2}$ is tight for Algorithm 1.*

Note that the example above also indicates that the classical greedy algorithm, which was proven with a ratio $1 - \frac{1}{e}$ for the classical MC problem due to Nemhauser et al. [23], can not have a ratio better than $\frac{1}{2}$ for maxSAF.

C. A RANDOMIZED LP ROUNDING ALGORITHM

In the subsection, we will develop an approximation algorithm by employing the randomized LP rounding technique against LP (2), improving the ratio 0.5 of the primal-dual algorithm to $1 - \frac{1}{e}$. Note that the ratio is already the best possible according to Theorem 6.

The key idea of our LP randomized rounding algorithm is first to compute an optimum solution to LP (2) and then to round the variables accordingly. It is known that an optimum solution to LP (2) can be computed in polynomial time [14]. So it remains only to show the method rounds an optimal (but fractional) solution of LP (2), say $(\boldsymbol{x}^*, \boldsymbol{y}^*)$, to an integral solution of maxSAF. However, there exist two difficulties for the rounding: The first is that, the two vectors of variables $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ are not independent with each other; the second, even inside the vector $\boldsymbol{x}^*$, two variables x_l and $x_{l'}$ can also be dependent with each other. E.g., assuming that S_l and $S_{l'}$ both belong to an identical $\mathcal{G}$, then if $x_l = 1$, $x_{l'}$ must be 0 according to Constraint (2) as in ILP (1). For the first difficulty, our algorithm will only round x_l with probability, and set y_i to 0 or 1 according to the value of x_l for which S_l contains the i^{th} POI. For the second, we will model rounding x_l as picking exactly one route for each user. However, which route to pick is depending on a probability proportional to x_l. Therefore, the rounding step of our algorithm is mainly as: For each user j, select a route l from $\mathcal{G}_j$ that S_l is selected with a probability proportional to x_l. Then the formal layout of the whole algorithm is as in Algorithm 2.

Algorithm 2 A randomized algorithm for maxSAF.

Input: A set of POIs $P = \{1, \ldots, m\}$ where the i^{th} POI is associated with a weight w_i, and U a set of users that $j \in U$ is associated with R_j and $\mathcal{G}_j$, where $\mathcal{G}_j$ is a collection of sets of POIs;

Output: $\mathcal{S}'$ a solution to *maxSAF*.

1: $\mathcal{S}' := \emptyset$;
2: Solve *LP (2)* against the instance of *maxSAF* by Karmakar's algorithm [14], and obtain an optimal solution $(\boldsymbol{x}^*, \boldsymbol{y}^*)$;
3: **For** each $j \in U$ **do**
4: Select a set S from $\mathcal{G}_j$, such that the probability of $S = S_l$ is proportional to $x^*_l \in \boldsymbol{x}$, i.e. $\frac{x^*_l}{\sum_{l \in R_j} x^*_l}$;
5: $\mathcal{S}' := \mathcal{S}' \cup \{S\}$;
6: **EndFor**
7: Return $\mathcal{S}'$.

Apparently, the runtime of Algorithm 2 is dominated by the time of solving LP (3), which is $O((q+m+n)^{3.5}L)$ by employing Karmakar's algorithm [14], where m, n and q are respectively the numbers of POIs, users and proposed routes, L is the length of the LP input in bits. So we have the following Lemma for the time complexity of Algorithm 2:

Lemma 13. *Algorithm 2 runs in time $O((q+m+n)^{3.5}L)$.*

Theorem 14. *Algorithm 2 is a randomized factor-$(1 - \frac{1}{e})$ approximation algorithm for maxSAF.*

Proof: Because $\frac{x^*_l}{\sum_{l \in R_j} x^*_l}$ is the probability that S_l is selected and $\sum_{l \in R_j} x^*_l \leq 1$, S_l is selected with probability not less than x^*_l. Then the probability of the i^{th} POI not being covered is:

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/ACCESS.2018.2868931, IEEE Access

$$\begin{aligned} P[\text{the } i^{th} \text{ POI not covered}] &\leq \prod_{l:\, i \in S_l} (1 - x_l^*) \\ &\leq \left(\frac{\sum_{l:\, i \in S_l} (1 - x_l^*)}{h_i} \right)^{h_i} \\ &= \left(1 - \frac{\sum_{l:\, i \in S_l} x_l^*}{h_i} \right)^{h_i} \\ &\leq \left(1 - \frac{y_i^*}{h_i} \right)^{h_i} \end{aligned}$$

where h_i is the number of sets containing i. The first inequality is from the famous inequality of arithmetic and geometric mean, while the second equality and inequality can be obtained by simple calculation.

So the probability of the i^{th} POI being covered is:

$$P[\text{the } i^{th} \text{ POI is covered}] \geq 1 - \left(1 - \frac{y_i^*}{h_i} \right)^{h_i}.$$

Let $f(y_i^*) = 1 - \left(1 - \frac{y_i^*}{h_i}\right)^{h_i}$. By calculation, we have

$$1 - \frac{y_i^*}{h_i} \leq e^{-\frac{y_i^*}{h_i}}.$$

Then $f(y_i^*) = 1 - \left(1 - \frac{y_i^*}{h_i}\right)^{h_i} \geq 1 - \left(e^{-\frac{y_i^*}{h_i}}\right)^{h_i} = 1 - e^{-y_i^*}$, and $f(0) = 0$ and $f(1) = 1 - e^{-1}$. Further, we have $f''(y_i^*) \leq 0$ when $0 \leq y_i \leq 1$. So $f(y_i^*)$ is concave. Then we know that

$$f[\lambda x + (1 - \lambda) y] \geq \lambda f(x) + (1 - \lambda) f(y).$$

Let $x = 1$ and $y = 0$ for the above formula, then $f(\lambda) = f(\lambda + (1 - \lambda) \cdot 0) \geq \lambda f(1) + (1 - \lambda) f(0) = (1 - e^{-1})\lambda$. Therefore,

$$\begin{aligned} P[\text{the } i^{th} \text{ POI is covered}] &\geq f(y_i^*). \qquad (5) \\ &\geq y_i^* f(1) \\ &= y_i^* \left(1 - e^{-1}\right) \end{aligned}$$

Let w_i be the weight of the i^{th} POI. Apparently, the weight of an output of Algorithm 2, say SOL, is $w(SOL) = \sum_{i=1}^{m} \left(P[\text{the } i^{th} \text{ POI is covered}] \cdot w_i\right)$. Then combining Inequality (5) yields

$$\begin{aligned} w(SOL) &\geq \sum_{i=1}^{m} \left(1 - e^{-1}\right) y_i^* w_i \\ &= \left(1 - e^{-1}\right) \sum_{i=1}^{m} y_i^* w_i \\ &= (1 - e^{-1}) w(LP) \\ &\geq (1 - e^{-1}) w(OPT) \end{aligned}$$

where $w(LP)$ is the weight of an optimal solution to LP (2), which is not less than $w(OPT)$, the weight of an optimal solution to maxSAF. This completes the proof. ■

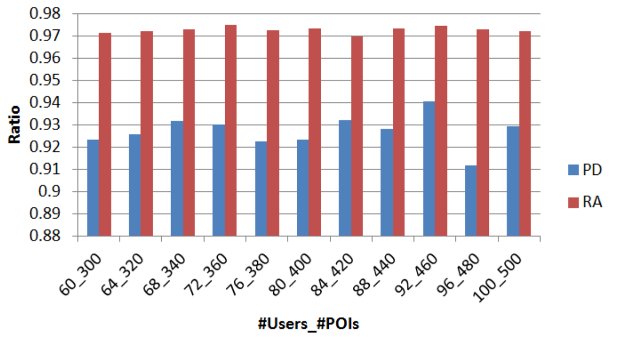


FIGURE 1: Practical approximation ratio of PD and RA for maxSAF in small scale.

D. SIMULATION EXPERIMENTS

In this subsection, for maxSAF we shall evaluate the practical performance of Algorithm 1 (the primal-dual algorithm, denoted as PD) and Algorithm 2 (the randomized algorithm, RA), by comparing their solution quality and runtime with each other, and also against the optimal exact algorithm (EA) from solving ILP (1). Also note that for the minMSCF problem, because our solutions Algorithm 3 and 4 have much more theoretical value rather than practical value, we will not evaluate them in the next section via simulation experiments.

Experimental Setup

The simulation experiments are carried out on a PC with Intel i5 4430 processor and 8G DDR3 memory and based in Windows 10 operating system. The algorithms are implemented using Python 2.7, in which we adopt the GLPK library[1] to solve ILP and LP.

Since solving ILP (i.e. EA) generally requires high runtime (typically exponential) which is not scalable, the simulation experiments are divided into two groups: small scale and large scale. For generated small scale datasets or data points as adopted in Figure 1 and left part of Figure 3, the number of users and POIs are respectively between 60-100 and 300-500. For the large scale as shown in Figure 2 and right part of Figure 3, the number of users and POIs are respectively in the ranges of 100-300 and 500-1500. POI data points are then allocated to users randomly, at the probability $\frac{(\text{the number of POIs})*5}{(\text{the number of user})^2}$. In the experiments, ten routes are generated for each user, where an allocated POI to a user is again randomly and uniformly distributed in three of the ten routes to ensure legitimate instances. The experiments generate 1000 instances for each size of maxSAF and the reported results below are averaged figures.

Comparison of Solution Quality

In this study, solutions from PD and RA are either compared with each other or compared against the optimal solutions

[1] https://www.gnu.org/software/glpk/

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/ACCESS.2018.2868931, IEEE Access

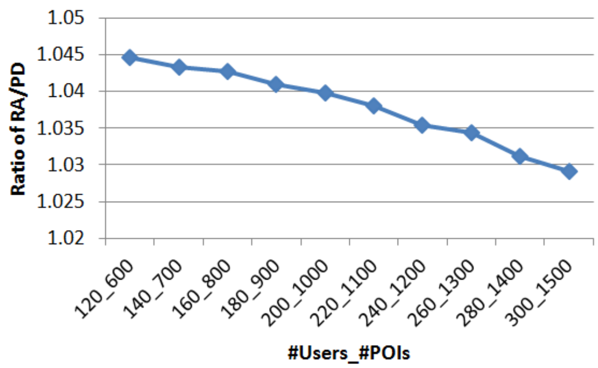


FIGURE 2: Ratio performance of *PD* vs *RA* for *maxSAF* in large scale.

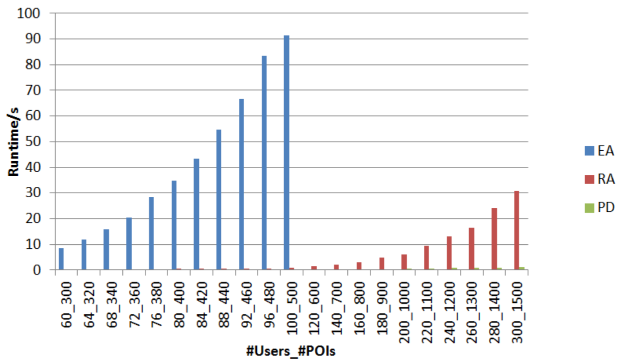


FIGURE 3: Runtime comparison of EA, RA and PD in small and large scale.

of EA, obtained by directly solving ILP (1) using the GLPK library. Figure 1 side-by-side compares the practical approximation ratios of PD and RA in small scale, i.e. the weight sum of the covered POIs output from PD and RA comparing to that of EA. As shown in Figure 1, RA has a slightly better coverage weight than PD, matching our theoretical ratio analysis, where RA and PD respectively reach up to 97.5% and 94% of the optimal solution. Another interesting finding is that these practical results overall are significantly better than the theoretical approximation ratios of RA and PD, which are respectively 0.63 and 0.5. With a little more thought, these results are nevertheless reasonable, because the theoretical ratios are for worst case instances while the experimental results are generally for average cases.

Further as shown in Figure 2, the case of a larger scale, the total weight (i.e. the solution value) of the covered POIs from RA is only slightly better than that of PD with up to relatively 1.045 times. Moreover, there is a clear trend that the performance gap between the two algorithms narrows down as the problem size increases.

Comparison of Runtime

In this study, the practical runtimes of EA, PD and RA are reported. The left part of Figure 3 compares the runtime of EA, RA and PD in small scale, while the right part compares only RA and PD in a larger scale as EA solver started to take too long to complete. For small scale data, the running time of RA is already significantly better than EA, and when the problem size increases, the gap between the two algorithms grows. In particular, at the point of 100 users and 500 POIs, EA is about 120 times slower than RA. Moreover, the runtime of PD is even much better (below 0.1 second) and harder to visualize in these small scale tests.

For the larger scale, the right part of Figure 3 depicts that the runtime of PD is much better than RA which is as expected from our theoretical analysis. This makes PD the fastest and probably the most preferable one among all three algorithms only with minimally sacrificed solution quality. Also the performance gap between PD and RA grows as the problem size increases: at the point of 300 users and 1500 POIs, RA takes more than 30 times of the runtime than PD.

IV. A DUAL-FITTING ALGORITHM FOR *MINMSCF*

In this section, we shall first give a greedy algorithm for minMSCF for the case that each user proposes one route. Then by employing the linear programming (LP) Dual-fitting technique and following a same line as in [14], we prove that the proposed algorithm has an approximation ratio of $O(\ln m)$, i.e. the solution output by the algorithm has a cost at most $O(\ln m)$ times of the optimum cost.

A. A GREEDY ALGORITHM FOR MINMSCF

Our algorithm is inspired by the classical greedy algorithm for set cover [13]. In general, the algorithm is composed by iterations, each of which selects a currently "best" set, until all the POIs are covered. It remains to defined a currently "best" set. Let $\hat{S}_j$ be the set of POIs that have not yet been δ-covered in S_j at the beginning of the an iteration. Then the average coverage cost of S_j exactly before the pth iteration is $\frac{c_j}{|\hat{S}_j|}$. Then, the algorithm is to repeatedly select S_j with minimum average coverage cost. Formally, the layout of the algorithm is as in Algorithm 3.

Lemma 15. *Algorithm 3 outputs a solution with a coverage cost at most $O(\ln m)$ times of that of an optimum solution.*

Here we can not simply extend the ratio proof for Set Cover with the main idea to sum up the cost of covering new POIs in iterations, because we do not know how to bound the cost of newly generated δ-covered POIs. Instead we propose a dual-fitting algorithm that is identical to Algorithm 3. By arguing its ratio being $O(\ln m)$, we equivalently prove Lemma 15.

Lemma 16. *Algorithm 3 runs in time $O(n^2 + mn)$.*

Proof: Apparently, the while-loop that starts from Step 2 of Algorithm 3 iterates for at most $O(n)$ times, since there are n sets (each for one user) in $\mathcal{S}$ and in each iteration the number of sets decreases at least one. Inside the while-loop, Step 6 takes at most $O(n)$ time to select the best S_l while the for-loop in Step 8 clearly takes $O(m)$ time

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/ACCESS.2018.2868931, IEEE Access

Algorithm 3 A greedy algorithm for *minMSCF*.

Input: A set of POIs $P = \{1, \dots, m\}$, a set of users $U = \{1, \dots, n\}$ in which each $j \in U$ is associated with a cost c_j, a route set of POIs S_j, an integer $\delta > 0$, and $\mathcal{S} = \cup_{j \in U} \mathcal{G}_j = \{S_j | j \in U\}$;

Output: $\mathcal{S}'$, a solution to *minMSCF*.

1: $\mathcal{S}' := \emptyset$, $p_1 = \dots = p_n = 0$;
/* p_i is the number of sets in $\mathcal{S}'$ containing i. */
2: **While** $P \neq \emptyset$ **do**
/* P contains the POIs that are not yet δ-covered. */
3: **If** $\mathcal{S} = \emptyset$ **then**
4: Return "Infeasible";
5: **Endif**
6: Select $S_j \in \mathcal{S}$ with *minimum* $\frac{c_j}{|S_j|}$;
7: $\mathcal{S}' := \mathcal{S}' \cup \{S_j\}$, and $\mathcal{S} := \mathcal{S} \setminus \{S_j\}$;
8: **For** each $i \in S_j$ **do**
9: $p_i := p_i + 1$;
10: **Endfor**
11: **For** each $i \in S_j$ **do**
12: **If** $p_i \geq \delta$ **then** /* i is δ-covered. */
13: $P := P \setminus \{i\}$;
14: **For** each $S \neq S_j$ in $\mathcal{S}$ that contains i **do**
15: $S := S \setminus \{i\}$;
16: **Endfor**
17: **Endif**
18: **Endfor**
19: **Endwhile**
20: Return $\mathcal{S}'$.

as $|S_j| = O(m)$. Counting the while-loop, Step 6 and 8 take $O(n^2)$ and $O(nm)$ time, respectively. It remains to calculate the time for executing Steps 11-18. Note that each POI i can be removed from P for only once, so the for-loop starting from Step 14 is executed for at most $O(m)$ times, each of which takes $O(n)$ time. That is, Steps 11-18 take $O(mn)$ time, equivalent to the total time consumed by Step 8. Therefore, the time complexity of the algorithm is $O(n^2 + mn)$. ■

B. THE RATIO PROOF VIA DUAL-FITTING

Denoting by $x_j \in \{0, 1\}$ whether S_j is selected, we have the LP relaxation for minMSCF as LP (4) below:

$$\begin{array}{lll} \min & \sum_{j=1}^{n} x_j c_j & \\ s.t. & \sum_{j:\, i \in S_j} x_j \geq \delta \quad i \in P & (6) \\ & x_j \geq 0 & \end{array}$$

Note that Constraint (6) guarantees that each POI will be covered by at least δ sets. Then we immediately have its dual LP (5):

$$\begin{array}{llll} \max & \sum_{i=1}^{m} \delta y_i & & \\ s.t. & \sum_{i: i \in S_j} y_i \leq c_j & j \in U & (7) \\ & y_i \geq 0 & i \in P & (8) \end{array}$$

Algorithm 4 A Dual-Fitting algorithm for *minMSCF*.

Input: $U = \{1, \dots, n\}$, $P = \{1, \dots, m\}$, an integer $\delta > 0$, and $\mathcal{S} = \{S_j | j \in U\}$where $S_j \subseteq U$ is with cost c_j;

Output: $\boldsymbol{x}$, an invalid dual solution that is corresponding to a solution to *minMSCF*.

1: $\mathcal{S}' := \emptyset$;
/* u_i is the number of sets in $\mathcal{S}'$ containing i. */
2: **While** $P \neq \emptyset$ **do**
/* P contains the POIs that are not δ-covered. */
3: Select $S_j \in \mathcal{S}$ with *minimum* $\frac{c_j}{|S_j|}$;
4: $\mathcal{S}' := \mathcal{S}' \cup \{S_j\}$, and $\mathcal{S} := \mathcal{S} \setminus \{S_j\}$;
5: Set $y_i = \frac{c_j}{|S_j|}$ for each $i \in S_j$;
6: **For** each $i \in S_j$ **do**
/* The post-processing after the selection of S_j. */
7: **If** i is δ-covered **then**
8: $P := P \setminus \{i\}$;
9: **For** each $S \in \mathcal{S}$ that contains i **do**
10: $S := S \setminus \{i\}$;
11: **Endfor**
12: **Endif**
13: **Endfor**
14: **Endwhile**

The following proposition is the start point of the dual-fitting algorithm, and known can be easily obtained by the theory of linear programming:

Proposition 17. *For the primal LP* (4) *and the dual LP* (5)*, we have* $\min \sum_{j=1}^{n} x_j c_j \geq \max \sum_{i=1}^{m} \delta y_i$.

Following the framework of the dual fitting method, we will first construct an invalid solution to the dual LP by a greedy algorithm; then we show that a feasible solution can be obtained by shrinking the solution within a bounded times. Then by Proposition 17, the shrunk solution is not larger than an optimal solution of the primal, and hence we get the ratio of the algorithm.

For the first task, we will show how to produce an invalid solution alongside with Algorithm 3. Let $\hat{U}$ be the set of indicators of the unselected sets. Then initially we have $\hat{U} = \{1, \dots, m\}$. Let $\hat{S}_j$ be the set of elements in S_j that currently have not yet been δ-covered. Then the assignment is, each time when $\hat{S}_j$ with the minimum $\frac{c_j}{|\hat{S}_j|}$ is selected (as in Algorithm 3), we set $y_i = \frac{c_j}{|\hat{S}_j|}$. The process of producing an invalid solution to the dual is formally as in Algorithm 4.

Note that the produced solution $\boldsymbol{y}$ of Algorithm 4 can be infeasible against the dual, since it might violate Constraint (7). So to prove the ratio, we show that by shrinking $\boldsymbol{y}$ by $O(\ln m)$ times, a new solution $\boldsymbol{y}'$ can be obtained from i.e. $y'_i = \frac{y_i}{\ln m}$, $\forall i$, is a dual feasible solution.

Theorem 18. *The dual solution* $\boldsymbol{y}'$ *satisfies Constraint (7).*

Proof: Assume S_j is picked in the pth iteration, and T_h, $1 \leq h \leq p$, is the set of the POIs in S_j which was δ-covered exactly after the hth iteration but not before. By the definition of T_h, when each POI $i \in T_h$ is eventually δ-covered by picking S_h, we have $y_i = \frac{c_h}{|\hat{S}_h|}$. Then because of the minimality rule over the ratio $\frac{c_h}{|\hat{S}_h|}$ during picking S_h, we have $\frac{c_h}{|\hat{S}_h|} \le \frac{c_j}{|S_j| - \sum_{l=1}^{h-1}|T_l|}$ where $\sum_{l=1}^{h-1}|T_l|$ is the number of POIs of S_j that are already k-covered before the hth iteration. Therefore,

$$\begin{aligned}\sum_{i:i\in S_j} y_i &= \sum_{h=1}^{p} |T_h| \frac{c_h}{|\hat{S}_h|} \\ &\le \sum_{h=1}^{p} |T_h| \frac{c_j}{|S_j| - \sum_{l=1}^{h-1}|T_l|} \\ &\le c_j \sum_{h=1}^{p} \frac{|T_h|}{|S_j| - \sum_{l=1}^{h-1}|T_l|}.\end{aligned}$$

By the definition of definite integration, we get

$$\sum_{h=1}^{p} \frac{|T_h|}{|S_j| - \sum_{l=1}^{h-1}|T_l|} \le \int_1^{|S_j|} \frac{1}{z} = \ln |S_j|.$$

So $\sum_{i:i\in S_j} y_i \le c_j \ln |S_j|$ and hence

$$\sum_{i:i\in S_j} y_i' = \sum_{i:i\in S_j} \frac{y_i}{\ln m} \le \frac{c_j \ln |S_j|}{\ln m} \le c_j.$$

Therefore, the solution $\boldsymbol{y}'$ satisfies Constraint (7). This completes the proof. ■

By Proposition 17, for the feasible solution $\boldsymbol{y}'$, we have $\sum_{i=1}^{m} k y_i' \le \sum_{j=1}^{n} x_j c_j$. Then combining $\sum_{i=1}^{m} k y_i' = \frac{1}{\ln m}\sum_{i=1}^{m} k y_i$ immediately yields $\sum_{i=1}^{m} k y_i \le \ln m \sum_{j=1}^{n} x_j c_j$, and eventually proves the correctness of Lemma 15. It is worth noting that the ratio is already tight for minMSCF, since the set cover problem which can be embedded in this minMSCF special case admits no approximation ratio better than $O(\ln m)$ unless $\mathcal{P} = \mathcal{NP}$ [6].

V. CONCLUSION

In this paper, we proposed the maximum Sweep Assignment problem with Flexibility (maxSAF) and resolved its hardness, i.e. it is $\mathcal{NP}$-complete even to decide if an instance of maxSAF contains a complete POI cover. To solve this problem, we first devised an approximation algorithm with a time complexity $O(mnq)$ and approximation ratio $\frac{1}{2}$ and then improved the factor to the tight ratio $1 - \frac{1}{e}$ with the runtime growing to $O((m+n+q)^{3.5}L)$ instead, where q is the number of routes and L is the length of input in bits. To complement these theoretical results, we also evaluated our algorithms via computer experiments to conclude their significant runtime gains in practice. In addition, we proposed the minimum Multiple Sweep Coverage problem with Flexibility (minMSCF) to encourage collecting higher quality sensor data. We first proved its general case does not admit any non-trivial approximation and then for its special case where each user proposes one route of POIs, we presented a greedy algorithm with time complexity $O(n^2 + mn)$ and theoretically analyzed that it is with a logarithmic approximation factor.

REFERENCES

[1] Reuven Bar-Yehuda and Shimon Even. A linear-time approximation algorithm for the weighted vertex cover problem. Journal of Algorithms, 2(2):198–203, 1981.

[2] G. Cardone, L. Foschini, P. Bellavista, A. Corradi, C. Borcea, M. Talasila, and R. Curtmola. Fostering participaction in smart cities: a geo-social crowdsensing platform. IEEE Communications Magazine, 51(6):112–119, 2013.

[3] Howie Choset. Coverage for robotics – a survey of recent results. Annals of Mathematics and Artificial Intelligence, 31(1):113–126, 2001.

[4] Stefan Dobrev, Stephane Durocher, Mohsen Eftekhari, Konstantinos Georgiou, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, Sunil Shende, and Jorge Urrutia. Complexity of barrier coverage with relocatable sensors in the plane. Theoretical Computer Science, 579:64–73, 2015.

[5] Rong-chii Duh and Martin Fürer. Approximation of k-set cover by semi-local optimization. In Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pages 256–264. ACM, 1997.

[6] Uriel Feige. A threshold of ln n for approximating set cover. Journal of the ACM (JACM), 45(4):634–652, 1998.

[7] B. Gorain and P. S. Mandal. Line sweep coverage in wireless sensor networks. In 2014 Sixth International Conference on Communication Systems and Networks (COMSNETS), pages 1–6, 2014.

[8] Barun Gorain and Partha Sarathi Mandal. Point and area sweep coverage in wireless sensor networks. In 2013 11th International Symposium and Workshops on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), pages 140–145, 2013.

[9] Longkun Guo, Min Li, and Dachuan Xu. Approximation algorithms for maximum coverage with group budget constraints. In International Conference on Combinatorial Optimization and Applications, pages 362–376. Springer, 2017.

[10] S. Hachem, A. Pathak, and V. Issarny. Probabilistic registration for large-scale mobile participatory sensing. In 2013 IEEE International Conference on Pervasive Computing and Communications (PerCom), pages 132–140, 2013.

[11] Dorit S Hochbaum. Approximation algorithms for the set covering and vertex cover problems. SIAM Journal on computing, 11(3):555–556, 1982.

[12] Dorit S Hochbaum. Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems. In Approximation algorithms for NP-hard problems, pages 94–143. PWS Publishing Co., 1996.

[13] David S Johnson. Approximation algorithms for combinatorial problems. In Proceedings of the fifth annual ACM symposium on Theory of computing, pages 38–49. ACM, 1973.

[14] Bernhard Korte, Jens Vygen, B Korte, and J Vygen. Combinatorial optimization. Springer, 2012.

[15] Andreas Krause, Eric Horvitz, Aman Kansal, and Feng Zhao. Toward community sensing. In Proceedings of the 7th International Conference on Information Processing in Sensor Networks, IPSN '08, pages 481–492. IEEE Computer Society, 2008.

[16] Jongwuk Lee, Dongwon Lee, and Seung-won Hwang. Crowdk: Answering top-k queries with crowdsourcing. Information Sciences, 399:98–120, 2017.

[17] Junkun Li, Jiming Chen, and Ten H Lai. Energy-efficient intrusion detection with a barrier of probabilistic sensors. In INFOCOM, 2012 Proceedings IEEE, pages 118–126. IEEE, 2012.

[18] M. Li, W. Cheng, K. Liu, Y. He, X. Li, and X. Liao. Sweep coverage with mobile sensors. IEEE Transactions on Mobile Computing, 10(11):1534–1545, 2011.

[19] Shuangjuan Li and Hong Shen. Minimizing the maximum sensor movement for barrier coverage in the plane. In Computer Communications (INFOCOM), 2015 IEEE Conference on, pages 244–252. IEEE, 2015.

[20] Chi Harold Liu, Bo Zhang, Xin Su, Jian Ma, Wendong Wang, and Kin K. Leung. Energy-aware participant selection for smartphone-enabled mobile crowd sensing. IEEE Systems Journal, 11(3):1435–1446, 2017.

[21] Prashanth Mohan, Venkata N Padmanabhan, and Ramachandran Ramjee. Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In Proceedings of the 6th ACM conference on Embedded network sensor systems, pages 323–336. ACM, 2008.

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/ACCESS.2018.2868931, IEEE Access

[22] Min Mun, Sasank Reddy, Katie Shilton, Nathan Yau, Jeff Burke, Deborah Estrin, Mark Hansen, Eric Howard, Ruth West, and Péter Boda. Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services, MobiSys '09, pages 55–68. ACM, 2009.

[23] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—i. Mathematical Programming, 14(1):265–294, 1978.

[24] Christos Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. In Proceedings of the twentieth annual ACM symposium on Theory of computing, pages 229–234. ACM, 1988.

[25] Sasank Reddy, Deborah Estrin, and Mani Srivastava. Recruitment framework for participatory sensing data collections. In Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic, editors, Pervasive Computing, pages 138–155. Springer, 2010.

[26] Matthias Stevens and Ellie D'Hondt. Crowdsourcing of pollution data using smartphones. In Workshop on Ubiquitous Crowdsourcing, 2010.

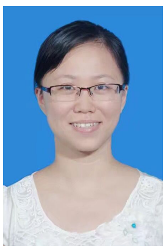
[27] Arvind Thiagarajan, Lenin Ravindranath, Katrina Leigh LaCurts, Samuel R. Madden, Hari Balakrishnan, Sivan Toledo, and Jakob Eriksson. Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, pages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

[28] Feng Wang, Liang Hu, Rui Sun, Jiejun Hu, and Kuo Zhao. Srmcs: A semantic-aware recommendation framework for mobile crowd sensing. Information Sciences, 433:333 – 345, 2018.

[29] M. Xi, K. Wu, Y. Qi, J. Zhao, Y. Liu, and M. Li. Run to potential: Sweep coverage in wireless sensor networks. In 2009 International Conference on Parallel Processing, pages 50–57, 2009.

[30] Guanqun Yang and Daji Qiao. Multi-round sensor deployment for guaranteed barrier coverage. In INFOCOM, pages 2462–2470, 2010.

[31] Zhiyong Yu, Jie Zhou, Wenzhong Guo, Longkun Guo, and Zhiwen Yu. Participant selection for t-sweep k-coverage crowd sensing tasks. World Wide Web, pages 1–18, 2017.



PEIHUANG HUANG received the B.S. and Master degrees in mathematics from Fujian Normal University, and is currently working for her ph.D. in computational intelligence. Her major research interest includes crowdsensing, sensor networks, and algorithm design.

WENXING ZHU received the B.Sc. degree in applied mathematics, and the M.Sc. and Ph.D. degrees in operations research and cybernetics from Shanghai University, Shanghai, China, in 1989, 1992, and 1996, respectively.

Since 1996, he has been with Fuzhou University, Fuzhou, China, where he has been a Professor with the Center for Discrete Mathematics and Theoretical Computer Science, since 2004. He has authored or co-authored over 40 scientific papers in refereed international journals, including the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, and the INFORMS Journal on Computing. His current research interests include optimization theory and methods, algorithms for VLSI physical design automation, and VLSI design for manufacturability. Dr. Zhu is currently an Associate Editor of the Journal Numerical Algebra, Control and Optimization, and the Journal Annals of Applied Mathematics

KEWEN LIAO completed a PhD at The University of Adelaide in 2014. He is currently a Research Fellow in Data Analytics at Swinburne University of Technology, with interests in data structures and algorithms for processing graph and time series data.

TIMOS SELLIS is Director of Swinburne's Data Science Research Institute. His research interests include big data, data streams, personalisation, data integration, and spatio- temporal database systems. Professor Sellis was elected IEEE Fellow in 2009 for his contributions to database query optimisation and spatial data management, and ACM Fellow in 2013 for his contributions to database query optimisation, spatial data management and data warehousing. In 2018 he was awarded the IEEE TCDE Impact Award, in recognition of his impact in the field and for contributions to database systems research and broadening the reach of data engineering research.

ZHIYONG YU is an associate professor at College of Mathematics and Computer Science, Fuzhou University, China, also affiliated with Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, and Key Laboratory of Spatial Data Mining and Information Sharing, Ministry of Education, China. He received his Ph.D. from Northwestern Polytechnical University, China in 2011. He was a visiting student at Kyoto University, Japan from 2007 to 2009 and a visiting researcher at TELECOM SudParis, France from 2012 to 2013. His current research interests include pervasive computing, mobile social networks, and crowd sensing.

LONGKUN GUO received the B.S. and ph.D. degrees in Computer Science from University of Science and Technology of China (USTC) in 2005 and 2011, respectively.

Longkun joined Fuzhou University in 2011. From 2015 to 2016, he was a research associate of the University of Adelaide, and is currently an associate professor and associate head of the department of computer science, College of Mathematics and Computer Science, Fuzhou University. His major research interest includes efficient algorithm design and computational complexity analysis for optimization problems in high performance computing systems and networks. He has published more than 30 academic articles in reputable journals/conferences such as Algorithmica, IEEE Transactions on Mobile computing, IJCAI, TPDS, SPAA, etc.