Crowdsourcing-based Spectrum Monitoring at A Large Geographical Scale

Yousi Lin
Virginia Tech
Blacksburg, United States
yousil94@vt.edu

Yuxian Ye
Virginia Tech
Blacksburg, United States
herexian@vt.edu

Yaling Yang
Virginia Tech
Blacksburg, United States
yyang8@vt.edu

***Abstract*—Spectrum monitoring is a powerful tool in dynamic spectrum access to help secondary users access the unused spectrum white space. The common approach for spectrum monitoring is to build infrastructures (e.g. spectrum observatories), which cost much money and manpower but have relatively low coverage. To aid in this, we propose a crowdsourcing based spectrum monitoring system for a large geographical area that leverages the power of masses of portable mobile devices. The system can accurately predict future spectrum utilization and intelligently schedule the spectrum monitoring tasks among mobile secondary users accordingly, so that the energy of mobile devices can be saved and more spectrum activities can be monitored. We also demonstrate our system's ability to capture not only the existing spectrum access patterns but also the unknown patterns where no historical spectrum information exist. The experiment shows that our spectrum monitoring system can obtain a high spectrum monitoring coverage and low energy consumption.**

***Index Terms*—DSA, crowdsourcing, spectrum monitoring**

I. Introduction

In traditional spectrum allocation policy, unshared access of fixed bands are given to licensed users [1]. However, this old allocation policy causes spectrum scarcity problem, where there is little spectrum left now to allocate to newly emerged communication demands. On the other hand, plenty of licensed bands are underutilized in both time and frequency domains, known as the spectrum white space. Hence, a new spectrum sharing paradigm, called dynamic spectrum access (DSA), is proposed to mitigate this problem. In DSA, incumbent spectrum users are called primary users (PUs) and the secondary users (SUs) can opportunistically access the spectrum white space as long as they do not trigger any harmful interference to the PUs' communications. In this way, DSA can utilize the considerable amount of underutilized spectrum [2].

One major way to search for the idle bands for SU access is spectrum monitoring. Spectrum monitoring is a powerful tool that help users effectively manage the frequencies to avoid improper usage and determine the sources of interference. There are some dedicated spectrum monitoring observatories and stations providing detailed spectrum occupancy information. These systems keep scanning the spectrum, and extract useful information, which will be finally stored in various spectrum databases. Spectrum occupancy pattern is a common type of the extracted information. Each pattern represents the spectrum utilization block in both time and frequency domain. It is a time-frequency representation of corresponding PU's behavior.

Nevertheless, there are two crucial disadvantages of the old systems and monitoring approaches. Firstly, today's spectrum observatories are not widely deployed and are mostly installed by government or well-capitalized enterprises due to the expensive and cumbersome hardware that is difficult to manipulate. Secondly, most of these systems merely scan the spectrum sequentially which is naive and inefficient.

To address the current limitations, we propose to leverage the power of mass on the concept of crowdsourcing [3]. Specifically, we let the mobile SUs to act both as users of available spectrum white space and moving spectrum monitoring devices. With a central controller coordinating a large amount of commodity secondary mobile devices, we can achieve low-cost spectrum monitoring at a large geographical scale.

Crowdsourcing based spectrum monitoring face two non-trivial problems: limited energy and dynamics in crowd distribution. Mobile users have their own daily usage routines. Spectrum monitoring functions should not interfere with the normal operation of other applications. Therefore, the energy of the mobile devices that can be spared on spectrum monitoring is limited. Also, one important characteristic of mobile users is mobility. The number of devices within an area that can be used to monitor the spectrum varies at different times and may lead to temporary insufficiency in local device number. Hence, the difficulty lies in how to achieve the highest spectrum monitoring coverage with the limited energy and changing number of mobile users.

We solve these challenges by leveraging the PU occupancy pattern. As shown in Figure 1, real spectrum access behaviors exhibit clear patterns. In this figure, a white block denotes a period of channel idle time, and the dark parts with some bright texture denote the behavior of active spectrum users. Thus, recorded past spectrum dataset not only can be used to extract past spectrum access patterns, but also can implicate the future dynamics in spectrum access. Thus, our spectrum monitoring system uses a smart scheduling algorithm that discovers the dynamic patterns of past spectrum activities to improve monitoring efficiency and reduce energy consumption within this system. Besides existing patterns, our system also includes an algorithm to intelligently watch for and learn new

This work was funded by the National Science Foundation under Grant No 1824494 and 1547366.

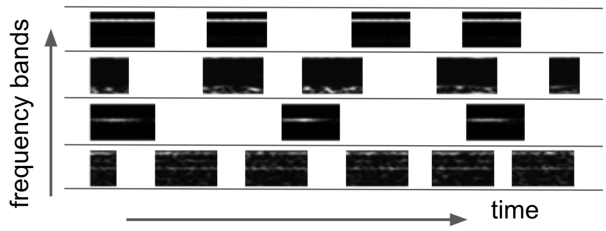

Fig. 1. From top to bottom, 4 examples of PU patterns from Cityscape Spectrum Observatory [4] used in our experiment.

patterns that have not been found in past history.

The rest of this paper is arranged as follows. Section II presents the related work. Section III explains our system model. Section IV and V describes how we use pattern information to intelligently assign monitoring tasks among SUs and how we discover new patterns. Section VI evaluate our schemes and finally section VII concludes our work.

II. Related Work

Mobile crowdsensing (MCS) is increasingly growing in popularity due to the extensive use of sensor-equipped mobile devices, and has been adopted in a diverse of sensing applications ranging from environment monitoring to human activities detection [5]. This paper mainly focuses on MCS works related to the real world spectrum monitoring. These works can be categorized into two aspects: (1) *Hardware implementation* [6], [7]: These existing works have built the hardware prototypes for distributed spectrum monitoring systems using commodity mobile devices. However, these papers do not address how to manage their systems for optimal monitoring performance. (2) *Spectrum monitoring based on fixed sensors* [4], [8]: These works assume spectrum sensors are dedicated and fixed devices. Their sensor coordination schemes are also simply letting each sensor to sequentially scanning the spectrum. An exception to this is a smart scheduling algorithm proposed in [9]. But this work is only designed for a single fixed sensor. Our work is the first that designs a crowdsoucing based spectrum monitoring system with intelligent task scheduling algorithms to coordinate among mass mobile devices (e.g. smartphones with spectrum monitoring capability).

To participate in crowdsourcing spectrum monitoring, mobile users would naturally expect certain payoff for the consumed time and energy. This situation requires effective incentive mechanisms. It is divided by [10] into the following categories: (1) Incentive schemes designed for collecting certain number of data samples through adjusting the reward [11], [12]. (2) Incentive schemes based on quality [13], [14]: in this kind of schemes, not only the quantity and cost of collected data but also the quality are taken into consideration. (3) Incentive schemes considering the QoI and credibility of the collected data [15], [16]. These existing incentive schemes for attracting users to participate in crowdsourcing tasks are complementary to our spectrum usage monitoring system. We can adopt any one of them in our system.

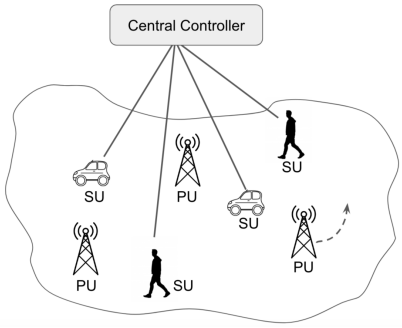

Fig. 2. Our system model.

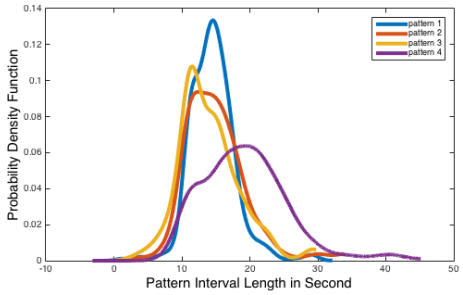

Fig. 3. PDFs of 4 kinds of pattern interval τ.

Other related existing works focus on how to classify PU patterns [17], [18] and how to predict future spectrum utilization [17], [19]. PU pattern classification and prediction are also required in our system, but they are not the emphasis of this paper. We apply a classic pattern classification method for PU pattern classification [20] and adopt an approach from [9] for future pattern prediction.

III. System Model

As in Figure 2, our system is built upon a model where there are multiple PUs and a large group of SUs scattered inside a large geographical region. The SUs are controlled by the central controller. The design of our model is based on two fundamental observations [9]: (1) PU signal has semi-regular pattern. The spectrum usage pattern represents how PU acts in both time and frequency domain. Because a PU's activities will not be completely random, we are able to find a certain distribution for the corresponding spectrum usage patterns. We refer a burst of a PU's transmission as one PU occurrence and the PU's pattern is essentially a sequence of its signal occurrences. In general, the intervals between consecutive occurrences of a PU have an uneven distribution. (2) PU's occurrence can be learned from history. The probability of a PU occurrence appearing within a certain time period can be calculated using the PU pattern's probability density function (PDF). Figure 3 shows the distributions of the occurrence intervals of 4 different PUs from real spectrum monitoring trace [4], which shows distributions similar to Gaussian distributions.

We assume some PUs are static and some PU can be mobile. All SUs are assumed to be mobile. According to

[7], there already exist several distributed spectrum monitoring prototypes built on smartphones. Hence, we can also assume all SUs have spectrum monitoring capabilities.

IV. Existing PU Pattern Monitoring

In this paper, we deal with two types of PU activity patterns. The first type is called *existing PU pattern*, which refers to ongoing PU transmission activities that also appeared in the past and hence were recorded. The second type is called *unknown PU pattern*, which refers to PU transmission activities that are not included in existing recorded dataset. In this section, we discuss our existing PU pattern monitoring scheme. In section V, we will outline unknown pattern monitoring.

Existing PU pattern monitoring leverages PU activity history in the spectrum databases for future spectrum monitoring. However, the PU activity information in these databases are all raw data. Therefore, a PU pattern classification procedure is required to identify existing PU patterns and compute the statistical distribution of occurrences in each pattern.

We extract PU patterns from existing PU activity datasets by the following two steps. First, we obtain the raw in-phase and quadrature (I/Q) information from the datasets and generate time-frequency occurrences. Second, we group PU occurrences into pattern groups based on similarity of the occurrences in frequency and time domain. The similarity is characterized by a 2D cross-correlation function [21] as:

$$P_1 \circ P_2(x,y) = \sum_{i=0}^{X}\sum_{j=0}^{Y} P_1(i,j)P_2(x+i,y+j), \quad (1)$$

where P_1, P_2 denote two occurrences, (X, Y) represent the discretized size of each occurrence at time and frequency domain, and the x, y denote the shifts of P_2 in time and frequency respectively. If two occurrences have a high cross-correlation, they are very similar in both time and frequency domain, thus they can be considered as in the same pattern.

For each existing pattern identified in the above process, we derive the statistical distribution of intervals between neighboring occurrences. Based on these distributions, a smart scheduling algorithm then assigns targeted monitoring tasks of existing PU patterns among SUs, so that we can reduce the energy consumption of PU monitoring. Essentially, our existing PU pattern monitoring algorithm leverages the fact that efficient monitoring of known patterns does not need to monitor every frequency band all the time. It can perform targeted monitoring only on time and spectrum range where the patterns have a chance to occur.

Three approaches for existing PU pattern monitoring are proposed as follows: The method named ideal optimal assignments is explained in Section IV-A. This method generates the optimal monitoring assignments under ideal and unrealistic assumptions. Its performance is treated as the upper bound for other approaches. A greedy optimization method is described in Section IV-B. Greedy optimization is a substitute to ideal optimal assignments with realistic assumptions. A heuristic algorithm is presented in Section IV-C. It can be regarded as a much faster alternative to greedy optimization.

TABLE I
Ideal optimal assignments

Given:	L_i^j, E_i for $i \in [1,N]$, $j \in [1,M]$ sd, T_{appear}, p_i, dt
Find:	A_i^j
Problem 1:	Minimize Energy Consumption
Minimize:	$\sum_{j=1}^{M}\sum_{i=1}^{N} A_i^j \times dt \times p_i$
Subject to:	(2), (3), (4), (5)
Problem 2:	Maximize Monitored Occurrences
Maximize:	$\sum_{j=1}^{M}\sum_{i=1}^{N} A_i^j$
Subject to:	(2), (3), (4), (6)

A. Ideal Optimal Assignments

The core of existing PU pattern monitoring is essentially the problem of coordinating SUs to achieve the optimal monitoring assignments based on PU occurrence time.

Consider a case where K PUs are scattered on a certain area, and N SUs are moving inside the area. K PUs generate a total number of M PU occurrences within a certain long period of time T (e.g. a day). The appearance time of the kth occurrence is denoted as T_{appear}^k and assume the indexes of the occurrences are sorted in ascending order of their occurrence time (i.e. $j < k \rightarrow T_{appear}^j < T_{appear}^k$). The energy budget in any SU i at time t for monitoring jobs is the energy left by all the other applications and is denoted as $E_i(t)$. For SU i ($i \in [1,N]$) and occurrence j ($j \in [1,M]$), define $A_i^j = 1(0)$ as (not) assigning SU i to detect occurrence j. L_i^j denotes the relation of SU i and occurrence j such that $L_i^j = 1(0)$ represents that SU i is (not) close enough to detect occurrence j. When monitoring the spectrum, SU devices require a short period of lead time to switch to a targeted frequency band, which is called a switching delay sd. Sensing duration dt is the time that an SU needs in sensing an occurrence of PU transmission. Monitoring power p_i is the power used when SU is active on spectrum monitoring.

Ideally, given all the parameters mentioned above, the problem is essentially to optimally assign the SUs to monitor all of the PU occurrences while minimizing the total energy consumed by monitoring assignments, which is formulated as the optimization Problem 1 in Table I. Sometimes due to the lack of SU devices or available energy for monitoring, the optimal formulation in Problem 1 becomes infeasible. If this happens, the goal of optimal assignment changes into finding an SU monitoring schedule that maximizes the number of PU occurrences that are monitored. This can be formulated as the optimal Problem 2 in Table I.

There are four constraints for optimal formulation in step 1. The first constraint specifies that each SU only has a limited monitoring range. During assignments, SU i cannot be assigned for occurrence j if occurrence j is out of SU i's geographical sensing range, which can be formulated as:

$$if\ L_i^j = 0, then\ A_i^j = 0,\ for\ i \in [1,N],\ j \in [1,M]. \quad (2)$$

The second constraint indicates that one SU can only be assigned at most one time to monitor one PU occurrence among the occurrences that appear within a period shorter than switching delay. This can be formulated as:

$$if\ j < k\ and\ T^k_{appear} - T^j_{appear} < sd + dt,\ then \sum_{j_1=j}^{k} A_i^{j_1} \leq 1, for\ i \in [1,N],\ j,\ k \in [1,M] \quad (3)$$

The third constraint needs to ensure that PU monitoring should not deplete energy for other SU normal activities. Thus, this constraint states that the algorithm can only use the leftover energy from other applications, as expressed as:

$$\sum_{j=1}^{j_1} A_i^j \times p_i \times dt \leq E_i(T^{j_1}_{appear} + dt), \quad for\ i \in [1,N],\ j_1 \in [1,M] \quad (4)$$

The fourth constraint ensures that each PU occurrence should be detected by at least one SU, as shown as:

$$\sum_{i=1}^{N} A_i^j \geq 1,\ \ for\ j \in [1,M] \quad (5)$$

When the formulation in Problem 1 is infeasible, we will execute the formulation in Problem 2. The constraints for Problem 2 are the same to Problem 1 except for constraint (5). Under Problem 2, the last constraint states that an SU should not be assigned to monitor the same PU occurrence twice. This can be formulated as:

$$\sum_{i=1}^{N} A_i^j \leq 1,\ \ for\ j \in [1,M] \quad (6)$$

B. Greedy Optimization With Uncertain Future

Note that the optimal assignment problem 1 and problem 2 in the last subsection only provide an upper limit on the performance of SU assignment algorithm. This is because they demand that the appearing time of every future occurrence and the leftover energy at every SUs are precisely known, which is impractical. Thus, in this subsection, we design a greedy optimal SU assignment strategy based on uncertain future as a practical solution for existing PU pattern monitoring. This greedy strategy can handle the uncertainty in the time and frequency information of PU occurrences and energy budget. Note that when there is uncertainty in PU occurrences, an SU may need to monitor a frequency band for a much longer amount of time than the sensing duration dt while listening and waiting for the possible PU occurrence to appear. Thus, the essence of our greedy approach is to greedily allocate the energy-limited monitoring time so that we can maximize our statistical coverage of PU occurrences in a short period T_{step} from the current time. The greedy computation is then repeated at the beginning of every T_{step} as time progresses. In the following, we describes the detail of this greedy approach.

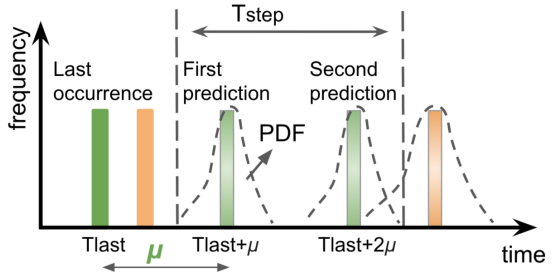

Fig. 4. Future occurrences of the pattern after T_{last} can be computed as PDF function with its mean shifted to $k\mu + T_{last}$. Here, different color represents occurrences in different patterns.

To maximize the monitoring tasks' statistical coverage of PU occurrences, the first step is to analyze the statistical distribution of PU occurrences. As in Figure 3, the occurrence time of each pattern will follow a certain distribution which can be learned from history. Probability that a PU occurrence appears within a certain time range can thus be computed based on the distribution. If we assign SU's limited monitoring time to span only those time ranges with high occurrence probability, we can maximize our coverage.

We define the time between two consecutive occurrences of a same pattern as the pattern interval τ. The statistical distribution of τ is computed based on this pattern's historical data. Its mean μ determines the mean occurrence interval, and standard deviation σ measures the variation of occurrences intervals in each pattern.

As in Figure 4, the distribution of τ can be used to generate a reliable future occurrence sequence in T_{step}. Assuming T_{last} is the last time where we observed a PU pattern's occurrence, distribution of the pattern's k_{th} future occurrence after T_{last} can be computed as the PDF function in Figure 5 with the mean shifted to $k\mu + T_{last}$. The cumulative density function (CDF) for each PU pattern is calculated based on PDF function. We then assign an SU monitoring time to maximize the probability of occurrence detection based on the CDFs.

Since the exact energy budget for monitoring (i.e. leftover energy of other applications) is also uncertain, a prediction of energy budget for spectrum monitoring in the current T_{step} is also needed to figure out the limit on monitoring time in each SU. In our greedy optimization, the prediction is based on profiles of mobile device usage. Using the usage information, at the current time t, we can predict the time when the next charging event will happen, denoted as T_{charge} , and other applications' future energy consumption till the next charging event, denoted as $E_{app}(t)$. We can also obtain the current battery energy $E_{battery}(t)$. Hence, the energy budget for each SU per T_{step} can be estimated by

$$E_i(t) = \frac{E^i_{battery}(t) - E^i_{app}(t)}{total\ number\ of\ T_{step}s\ before\ T^i_{charge}}, \quad (7)$$

We further suppose that SUs report their locations at the beginning of each T_{step}, and they will stay adjacent to this

location within T_{step}. Thus, the relation L_i^j between SU i and occurrence j can be computed using the location information.

Now given the above parameter definitions and assumptions, the goal of our greedy optimization can be formulated as finding the optimal start and end points for each monitoring task assignment happening within T_{step}. This greedy optimization problem can be formulated as in Table II, where ${t_0}_i^j$ and ${t_1}_i^j$ are defined as the start and finish time of an SU i's monitoring task on occurrence j respectively. Here, the objective function is essentially the statistical coverage of PU occurrences, which is defined as the summation of probabilities of capturing the occurrences that appear in T_{step}.

TABLE II
GREEDY OPTIMIZATION PROBLEM WITH UNCERTAIN FUTURE

Given:	L_i^j, CDF_j, E_i for $i \in [1, N]$, $j \in [1, M]$ sd, T_{step}, p
Find:	A_i^j, ${t_0}_i^j$, ${t_1}_i^j$
Maximize:	$\sum_{j=1}^{M} \sum_{i=1}^{N} A_i^j \times CDF_j({t_1}_i^j - {t_0}_i^j)$
Subject to:	(2),(8),(9), (10),(11),(12)

There are six constraints in the greedy optimization in Table II. The first and the second one capture the sensing range of each SU as in equation (2) and (8):

$$if\ L_i^j = 0,\ then\ {t_0}_i^j = 0\ and\ {t_1}_i^j = 0,\ for\ i \in [1, N],\ j \in [1, M] \tag{8}$$

The third constraint prevents overlapping of sensing tasks on the same PU occurrence to prevent wasting energy, as shown as:

$$if\ i \neq k,\ then\ {t_0}_i^j - {t_1}_k^j \geq 0\ or\ {t_0}_k^j - {t_1}_i^j \geq 0,\ for\ i,\ k \in [1, N],\ j \in [1, M] \tag{9}$$

The fourth constraint ensures that an SU is not assigned to two monitoring tasks at the same time and also the inter-task interval for an SU should be larger than the switching delay. This constraint is formulated as:

$$if\ j \neq k,\ then\ {t_0}_i^j - {t_1}_i^k \geq sd\ or\ {t_0}_i^k - {t_1}_i^j \geq sd,\ for\ i \in [1, N],\ j,\ k \in [1, M] \tag{10}$$

The fifth constraint represents the relation between ${t_0}_i^j$ and ${t_1}_i^j$. When assigning SU i to detect pattern j, ${t_0}_i^j$ should be smaller than ${t_1}_i^j$, and they should both be within the range of T_{step}, which can be expressed as:

$$if\ A_i^j = 1,\ then\ 0 \leq {t_0}_i^j < {t_1}_i^j \leq T_{step},\ for\ i \in [1, N],\ j \in [1, M] \tag{11}$$

The last constraint is similar to (3), which is to ensure that the energy consumption of each SU cannot exceed the corresponding energy budget at the beginning of T_{step}. The formulation is shown as:

$$\sum_{j=1}^{M} ({t_1}_i^j - {t_0}_i^j) \times p_i \leq E_i, for\ i \in [1, N], \tag{12}$$

where E_i is computed by equation (7).

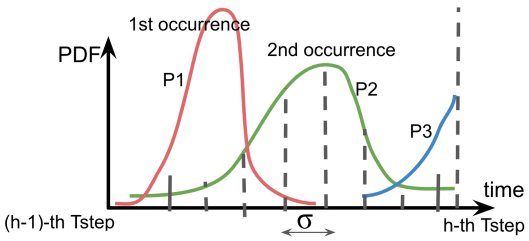

Fig. 5. An example showing how we divide the PDF into 7 portions. The interval between 2 consecutive dash lines are a portion of σ.

C. Fast Heuristic Algorithm

The greedy optimization problem in Table II needs to be solved repeatedly at the beginning of every T_{step}. Unfortunately, it is a mixed integer programming problem, which is in general NP complete. Thus, solving it directly is impractical due to its high computational complexity and the potential of dealing with a large amount of SUs in a large geographical scale. Thus, we propose a heuristic algorithm as a substitute to directly solving the greedy optimization problem. In essence, the heuristic algorithm is an approach that applies a rougher but faster way of iteration to reach an quasi-optimal results for the greedy optimization problem. In greedy optimization method, we use the optimization techniques to find the assignments that attain the largest coverage in the imminent T_{step} time, while in the heuristic algorithm, we design an iteration method to quickly generate the assignments with limited loss in coverage for this T_{step} time.

Similar to greedy optimization, we run the heuristic algorithm before the beginning of each T_{step} time. The heuristic algorithm has two steps: step 1 is to identify segments as assignment units to simplify the PDF of each predicted occurrence; step 2 is the segment-based SU scheduling.

Step 1: In step 1, we first divide the PDF of a PU occurrence from the center into 7 *sensing segments*, and each segment has a length of a standard deviation σ of the PDF as shown in Figure 5. Assuming that PU occurrences follow a PDF similar to normal distribution, which has been validated in analysis of PU behaviors as shown in Figure 3, the probability that an occurrence appears outside of the 7σ range can be approximated as 0. Each segment is the smallest time unit considered in our scheduling algorithm. Given the energy that should be reserved for each SU at each T_{step}, the goal of our scheduling algorithm is to properly utilize the energy within each T_{step} to monitor as much segments as possible for all pattern occurrences in T_{step}. If each segment of every occurrence has an assigned SU to monitor the corresponding PU's activity, then all PU activities will almost surely be captured by our scheduling algorithm.

Step 2: In step 2, we assign SU's monitoring jobs in unit of segment. There are *two scheduling cases* for each segment: (1) If there exists any SU that is capable of monitoring the PU for the entire sensing segment, the task of this segment will be allocated to a single SU. (2) For the segment with no feasible

assignment, we will use best effort assignment strategy.

For both cases, we first compute the relation L_i^j at the beginning of T_{step}. Based on L_i^j, we can obtain a list of SUs which are close enough to detect a PU j's signal. For each segment of a PU occurrence, we further narrow down the SU list to only include those with sufficient energy and have no other on-going monitoring assignment to cover the entire segment. If the resulted list is not empty, we have case 1 and use a metric $reward_{i,j}^k$ for picking the best SU from the list to monitor the target segment. The higher the value of $reward_{i,j}^k$ is, the more preferable that SU i is for monitoring segment k of PU j's occurrence. When we cannot find any SU that has adequate energy or time to sense an entire segment, we have case 2 and will use best effort strategy. In this case, SU with the largest reward metric $reward_{i,j}^k$ will provide its best effort to cover as much of the segment as possible, and then leave the rest of the segment to the SU with the second largest reward and so forth, until the segment is entirely covered, or there is no assignable SU left. Essentially, the two cases described above ensures that we first try to assign a single SU to monitor an entire segment before we resort to the best effort assignment that assigns monitoring tasks with finer time span. This preference for larger time unit in assignment is because an SU needs to pay the cost of switching delay. Thus, assignment of an SU to monitor a too small time period is not efficient in terms of monitoring time management.

For both cases, we use a metric $reward_{i,j}^k$ for picking the best SU from a list of SU candidates to be assigned to monitor the target segment. This $reward_{i,j}^k$ is designed to be a product of three factors as follows:

$$reward_{i,j}^k = W_{i,j}^k \times E_i \times value_i^j, \tag{13}$$

where $W_{i,j}^k$ denotes SU i's relative profit of sensing the segment k of PU j's occurrence comparing to the choices of monitoring the segments of other PUs' occurrences; $value_i^j$ denotes how valuable is SU i's potential monitoring service for occurrence j comparing to the other possible choices of SUs for this monitoring job. The calculation for energy budget E_i follows equation (7), identical to that of greedy optimization. $W_{i,j}^k$ is computed as below:

$$w_{i,j}^k = prob_j^k \times L_i^j;\ W_{i,j}^k = w_{i,j}^k - maxW_j^k if\ w_{i,j}^k > 0 \tag{14}$$

where $prob_j^k$ is the probability that an PU j's occurrence appears in segment k , $w_{i,j}^k$ denotes the absolute profit (i.e. chance of capturing an occurrence) if we assign SU i to monitor segment k of PU j, and $maxW_j^k$ is defined as the largest w value that can be obtained by SU i if SU i spends segment k's time to monitor other PUs occurrences instead of PU j's occurrence. Mathematically,

$$maxW_j^k = \max_{l \neq j,m} w_{i,l}^m L_i^l O_{j,l}^{k,m}, \forall i,j \text{ satisfying } L_i^j = 1, \tag{15}$$

where $O_{j,l}^{k,m} = 1$ if segment k of PU j's occurrence time overlaps with segment m of PU l's occurrence time and $O_{j,l}^{k,m} = 0$ for other cases. Essentially $maxW_j^k$ is a loss that should be taken into account for relative profit calculation because an SU cannot simultaneously monitor multiple PUs. If an SU is allocated to monitor a segment of one PU, it will lose the probability of capturing occurrences of other PUs.

The design of the $value_i^j$ is based on the following rationale. Assuming an SU i is able to hear more than one PUs' signal, the urgency of mandating SU i to monitor a particular PU j's segment is related to the number of SUs within the area that can also sense PU j's signal. If many SUs can detect PU j's signal, then it is not a big issue to arrange SU i to sense other PUs' signal since it is easy to find alternative SUs for the monitoring task of PU j. Thus, the $value_i^j$ will be small in this case. Based on this reasoning, we define:

$$value_i^j = 1 - \frac{X_j}{\sum_{l \in S(i)} X_l} \tag{16}$$

where X_j denotes the number of SUs that can hear PU j's signal and $S(i)$ is defined as the set of PUs whose signal can be heard by SU i.

At the beginning of each T_{step}, we determine which case will be encountered and then perform the reward metrics computation. The procedure will be repeated for each T_{step}.

V. Unknown PU Pattern Monitoring

So far, we only mandated SUs to monitor the PU patterns occurred in the past. However, historical information will grow inaccurate as time goes by due to PUs' spectrum access behavior change and mobility, which will finally degrade the performance of our proposed system. To keep our system updated with PU dynamics, we also need to discover new PU patterns, which is defined as pattern exploration in [9].

Using **unknown pattern** to represent new PU patterns that are not included in our existing pattern dataset, our design aims to watch as many unknown patterns as possible while stably monitoring the occurrences of existing patterns.

A. Heuristic Algorithm based on SU Location

Suppose that after finishing existing pattern detection at the beginning of each T_{step}, some SUs may still have extra remaining energy and idle time to be utilized for detecting unknown PU occurrences. Optimal assignments for this scenario can be generated using the same optimization formulation as in Table I of Section IV-A, except that occurrence i in this formulation means unknown pattern occurrence i. Again, the optimal assignments strategy assumes an ideal situation where pattern occurrence sequence and the locations of all SUs and PUs are known beforehand.

Due to the optimal assignments' unrealistic assumptions, we propose another heuristic algorithm for unknown PU pattern monitoring. The core of this algorithm lies in: (1) Without any prior information on new PU patterns, SUs randomly scan the spectrum to capture new PU occurrences. (2) If any new occurrences are detected, the central controller will quickly model the distribution of intervals between new occurrences, so that future monitoring job assignments on new PU patterns can be more targeted and accurate.

The design of the heuristic algorithm has five parts: grouping SUs based on locations, deciding SUs' monitoring schedule, random monitoring, targeted monitoring, and updating patterns in existing pattern sets.

*1) **Group SUs based on locations**:* Intuitively, different SUs will have high correlation in sensing outputs if they are geographically close enough. We define the SUs whose sensing ranges have adequate overlap as one group, closeby SUs belonging to the same group are considered to monitor the same area. We treat the SUs in a group as an aggregate for monitoring jobs to reduce energy consumption of each SU.

There is a trade-off between energy consumption and sensing accuracy when choosing the overlap threshold. On one hand, smaller overlap threshold may result in a loss of sensing coverage since the algorithm may not detect new PU signals that only occurred in the non-overlapping area. On the other hand, the number of SUs in a group decreases as the threshold increases, which means an SU will need to spend more energy and time to handle this group's monitoring tasks. In this paper, we set the overlap threshold to 80% of a device's sensing range. This value is adjustable based on the desired level of trade-off between energy consumption and sensing accuracy.

*2) **Decide SUs monitoring schedule**:* For each group, we need to decide the monitoring schedule of SUs inside it to maximize the aggregated monitoring time. There are two problems: How much usable time will this group have? How much time and energy each SU should contribute to this group? For each T_{step}, the assignable time of a group is determined by the idle time and energy of its SUs. Our goal is to assure as much usable time as possible for that group. Therefore, we set a metric $R_{user\_in\_group}$ as the criterion to allocate SUs' idle periods to the groups. A SU is more valuable to its group if it has larger $R_{user\_in\_group}$ compared to other SUs. This metric R is determined by the sensor priority Sp and energy budget E_i of SU i as:

$$R_{user\_in\_group} = Sp_i \times E_i \tag{17}$$

Sensor priority is a parameter characterizing the importance of an SU's monitoring service in a group, which is related to the number of SUs in that group. If a group contains many SUs, it is trivial not assigning a particular SU to sense an occurrence since it is easy to find alternative SUs for that monitoring job. The priority of SUs in a group is computed as the reciprocal of the group's SU number. If an SU is included in multiple groups, it will be treated as different devices as in computing the corresponding metrics. For example, if SU 1 is in both group A and B, suppose group A and B has 2 and 3 SUs respectively, the rewards of SU 1 in group A and B are calculated as $R_{1\_in\_A} = \frac{1}{2} \times E_1$ and $R_{1\_in\_B} = \frac{1}{3} \times E_1$.

In each step, all the values of $R_{user\_in\_group}$ are sorted in a descending order. Then, we allocate the idle time of each SU to its corresponding group following this list of $R_{user\_in\_group}$. Specifically, when taking an entry of $R_{i\_in\_A}$ in the list, we first check if T_{step} is totally covered in group A. If the answer is yes, we skip to check the next entry in the list. Otherwise, we allocate SU i's idle time for monitoring job in group A. The process continues until T_{step} is totally covered in every group or no usable time or energy of the SUs is left. Time segments of different SUs assigned to the same group should not be superposed as to reduce energy consumption.

*3) **Random monitoring**:* Note that there is no prior knowledge on where and when a new pattern will appear. Hence, we uniformly divide a groups' sensing time into multiple small pieces and randomly select a frequency band to sense during each time piece. If any unknown occurrences are found, we record the time and frequency of those occurrences, and the locations where the SUs stayed when detecting the new occurrences. We also cluster the occurrences into different patterns. This stage creates the unknown pattern dataset.

*4) **Targeted monitoring**:* After capturing some new PU occurrences, we can enter the targeted monitoring stage, where we compute the mean and deviation of each new pattern's intervals to model its statistic distribution.

When a new pattern just starts to occur, it is more likely that there is not enough occurrence observations to accurately characterize the new pattern's occurrence distribution. However, we can see from the observations in Figure 3 that the distributions of different pattern intervals are usually approximate Gaussian distributions. Therefore, when pattern samples are inadequate, we model the pattern intervals as Gaussian distributions using computed mean and deviation. If only two occurrences of a pattern are found, we use the interval between these two occurrences as the mean value, and choose the average deviation of all kinds of patterns within the dataset as the new pattern's predetermined deviation. If just one occurrence of a pattern is detected, we use both the average mean and deviation of all kinds of patterns as the preset values for modeling. The mean and deviation value will be updated after each T_{step}.

We will be able to leverage the rough pattern interval models to predict where and when other new occurrences of this PU pattern will appear in the next T_{step}. Heuristic algorithm in Section IV-C is adopted here to generate a group's sensing scheduling, except that the "worker" in this section is changed from SUs to groups.

After group scheduling is finished, we again do the random monitoring by uniformly dividing each group's remaining sensing time and randomly choosing a frequency band to monitor, in order to explore more unknown PU patterns.

*5) **Update patterns in existing pattern sets**:* There are two updates in existing pattern sets at the end of each T_{step}: (1) we will add newly discovered patterns to the pattern sets, and update the mean and deviation value of each pattern interval distribution. We will check if the location where we discovered a PU pattern is moving. If it is, the location information will be updated. (2) we will remove PU patterns that have not occurred for a long time from the sets.

Unknown pattern monitoring algorithm ensures that our system can start to function even when no historical data is provided. It gradually creates the pattern set and improve its monitoring intelligence based on the past data. New pattern

exploration is not isolated from our scheduling algorithm. The scheduling algorithm always uses up-to-date spectrum information obtained from both existing and unknown monitoring algorithms, thus is not wedded to a wrapped dataset.

B. Update Relations between PU and SU Locations

In all previous discussions of existing and unknown pattern monitoring, we assume that we know PU and SU locations so that we can determine if an SU can hear a PU's signal. An SU's location is easy to obtain from its location updates which are commonly performed by modern-day mobile devices. The location of PU, on the other hand, is not readily available . In this section, we will explain how we obtain and update the location information of PU.

During unknown PU pattern monitoring, the initial random assignment of monitoring jobs ensures that the occurrences of a new PU pattern will likely be detected by different SUs. These SUs are at different locations and the sensing results of these SUs can then be used to localize the PU through radio localization techniques such as trilateration and proximity [22].

When a new PU pattern becomes stably detected at predicted appearing times, it becomes an existing pattern.

At this stage, it is possible that only one SU is assigned by the existing PU pattern monitoring algorithm to monitor this pattern. Overtime, a mobile PU may drift away from its old location and the SU assigned by existing PU pattern monitoring algorithm using its old location may not hear the PU's signal anymore. If the unknown PU pattern monitoring algorithm in other SUs picked up the PU's signal before this happens, we know that this known PU has moved its location and can use radio localization techniques to recompute its new location. If the unknown PU pattern monitoring algorithm did not pick up the moving PU's signal before it went off the radar, this PU is considered disappeared. However, the PU is not lost permanently. It may soon get picked up by the unknown pattern detection algorithm at some SU as a new pattern and its new location will be identified again following the procedure of the previous paragraph.

VI. Experiment

In this section, we evaluate the performance of our crowdsourcing based spectrum monitoring scheme using simulations that are based on real world data. We did two sets of simulations. The first set examines the case where all patterns are stable and there is no unknown pattern, and the second set evaluates the cases where there are many unknown patterns. In the first set of simulations, we evaluated the greedy optimization method and heuristic algorithm and use the ideal optimal assignments as the benchmark. In the second simulation set, we evaluated the performance of heuristic algorithm with the ideal optimal assignments as a higher bound and a naive random detection method as a lower bound.

In our simulation, 44 types of real PU patterns in TV frequency bands are extracted from data in Cityscape Observatory spectrum databases [4], where 34 of them are used as existing PU patterns that are present from the beginning of an simulation, and another 10 patterns start in the middle of the simulation to act as unknown PU patterns. We also artificially makes 3 of the existing PU patterns to be from mobile PUs since the mobility of PUs should also be under consideration. The mobility of the PU is reflected in the changes of signal strength in SU's sensing results. We have to add artificial mobility of PU into the spectrum monitoring results because such sensed signal strength changes were not observed in existing spectrum dataset. Hence, we manually add the traces to some PUs to simulate the case where a PU is also moving. Figure 1 shows 4 examples of the PU patterns.

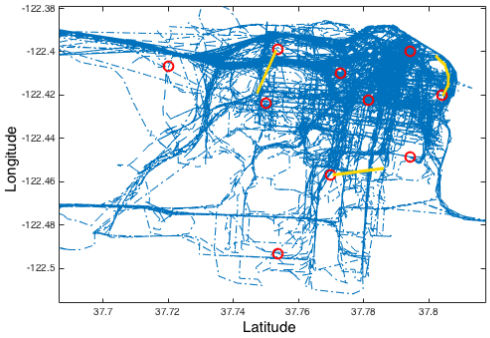

Fig. 6. A trace-driven simulation. (dash-dot lines: taxi movement trajectories. circles: PUs.) We also add movements (yellow lines) to 3 PUs.

We utilize 100 taxi trajectories from mobility traces dataset of taxi cabs in San Francisco [23] to represent SU mobility traces. Figure 6 shows a smaller scale illustration of our simulation setting with 60 taxi traces and 10 PU deployment.

To ensure our evaluation is based on realistic energy consumption on mobile devices, we utilized a Device Analyzer dataset from University of Cambridge [24] which contains: (1) times when a phone is turned on and off; (2) times at which the phone is charging; (3)the real time battery level and voltage, etc. Hence, we are able to compute from this dataset the real energy consumption of phone applications, which can be used for computing (7) in greedy optimization method and heuristic algorithm. For the energy consumption of spectrum sensing, according to [7], SU devices being active on sensing draws about 1.5W as the monitoring power and the sensing duration dt is set to be 5ms based on [9]. We also set the frequency switching delay sd in our simulation to 50ms based on the empirical measurement in [7].

A. Existing PU Pattern Monitoring

In this experiment, we simulated a crowdsourcing spectrum monitoring scenario of 2 hour duration, in which 34 PUs transmitting known patterns are deployed among 10 to 100 SU traces. A T_{step} lasts for 5 seconds. At each T_{step}, SUs report their locations to the central controller.

First, we compare existing pattern coverage, defined as

$$EPcoverage = \frac{DetectedExistingPatternNumber}{TotalExistingPatternNumber}. \quad (18)$$

among optimal assignments, greedy optimization method and fast heuristic algorithm in Figure 7. In the experiment, the SU number is increased from 10 to 100 to see the changes in the coverage. For each parameter setting, we run 10 independent random simulations and show the average results in Figure 7. From the figure, it can be observed that the coverage is apparently increasing with the growth of SU numbers. The performance of the greedy optimization method is slightly superior to that of the heuristic algorithm, since a heuristic algorithm cannot really guarantee an optimal solution.

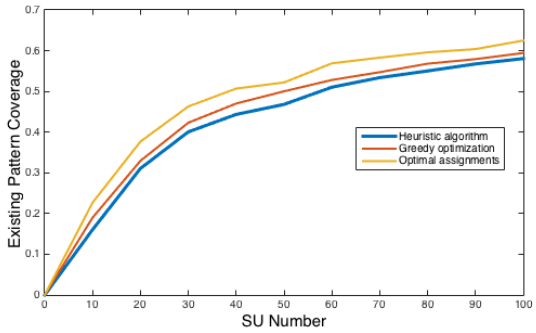

Fig. 7. Existing pattern coverage comparison. (Yellow line: Optimal assignments. Red line: Greedy optimization. Blue line: Heuristic algorithm)

Figure 8 further shows the total energy consumption of our spectrum monitoring algorithm in the above experiment, which is the summation of energy consumption of spectrum monitoring from all the SUS.

The total energy (presented in solid lines) consumed by the heuristic algorithm and greedy optimization are very close to each other, demonstrating the effectiveness of the heuristic algorithm. We also plot an energy ratio (presented in dash-dot lines) to show how much energy spectrum sensing really consumed against the total usable energy budget, which is defined as:

$$EnergyRatio = \frac{TotalUsedEnergyForSensing}{TotalUsableEnergyBudget} \quad (19)$$

From Figure 8, it can be seen that as the number of SUs increases, more total energy is used to get better coverage of PU signals, while the per SU's energy consumption actually decreases since more sensors are sharing the monitoring responsibility.

Figure 9 shows how the number of SUs available for spectrum monitoring jobs changes over time. An SU is available if it has extra energy to monitor spectrum in the next T_{step} time. We define the available SU ratio as:

$$AvailableSURatio = \frac{AvailableSUNumber}{TotalSUNumber}, \quad (20)$$

and record its changes in each T_{step}. As shown in Figure 9, the available SU ratio of heuristic algorithm fluctuates. This is due to the randomness in the energy budget, which is caused by the randomness in other applications' energy consumption. However, we do not observe drastic decrease in available SU numbers, because the energy budget used by

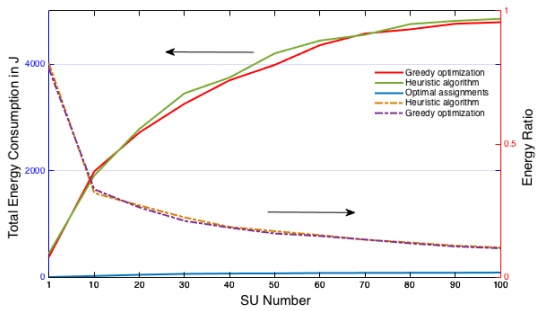

Fig. 8. Energy consumption evaluation. Left side shows the total energy consumption in J. Optimal assignments, greedy optimization and heuristic algorithm are presented using solid blue, red and green lines, respectively. Right side shows the energy ratio of heuristic algorithm and greedy optimization, which are presented using purple and orange dash-dot lines, respectively.

spectrum monitoring at each T_{step} is portioned based on an SU's past energy profile.

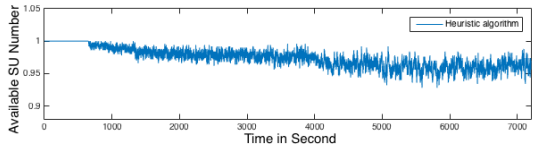

Fig. 9. Changes of available SU number in heuristic algorithm across time.

B. Unknown PU Pattern Monitoring

In this set of simulation, we add an additional 10 unknown patterns besides the 34 existing patterns. The other settings of the simulation is the same. As a reference, we run a random detection algorithm, which divides available sensing time of each SU uniformly into small time chunks of 50 ms in length. Each time chunk is assigned to monitor a random frequency band and the algorithm is not able to learn from history.

First, we show an unknown PU pattern coverage comparison over optimal assignments, SU location based heuristic algorithm and the random detection method. The coverage represents the method's capability of capturing new patterns. This unknown pattern coverage is defined as:

$$UPcoverage = \frac{DetectedUnknownPatternNumber}{TotalUnknownPatternNumber} \quad (21)$$

As in Figure 10, the output of random detection method is barely 25% percent of the output of our heuristic algorithm. This is because our heuristic algorithm can learn from history to assign the limited SU monitoring time on the most likely frequency and time of PU occurrence. There is still a gap between our heuristic algorithm and the ideal optimal assignments since the optimal assignment unrealistically assumes known future, while the heuristic algorithm is designed with uncertain future.

Since unknown PU pattern monitoring consumes energy, it may affect future energy budget for existing pattern monitoring. Thus, we compare existing pattern coverage of our

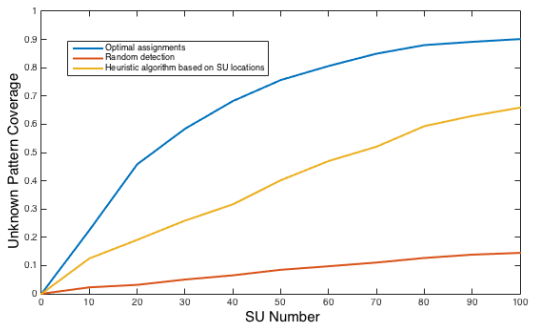

Fig. 10. Unknown pattern coverage comparison. Blue line: Optimal assignments. Yellow line: Heuristic algorithm based on SU location. Red line: Random detection.

heuristic algorithm with and without the unknown pattern monitoring part. Figure 11 shows the results. There is indeed a small coverage drop after adding unknown pattern monitoring. However, we believe that this small drop in existing pattern coverage is worthy to ensure our system can detect dynamic PU patterns as well as mobile PUs.

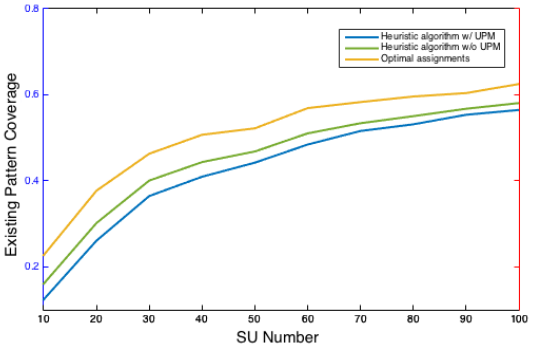

Fig. 11. A comparison showing how the unknown pattern monitoring (UPM) affects existing PU pattern coverage.

VII. Conclusion

This paper presents a novel crowdsourcing based spectrum monitoring system, which can intelligently schedule the monitoring tasks for masses of portable mobile devices, by leveraging the PU occupancy patterns. We propose three algorithms based on practical spectrum usage data, two for existing pattern monitoring and one for unknown pattern monitoring. Simulation results show that our system achieves a high spectrum monitoring coverage. Our system is also capable of running efficiently even when no historical spectrum information is available.

References

[1] M. Song, C. Xin, Y. Zhao, and X. Cheng, "Dynamic spectrum access: from cognitive radio to network radio," *IEEE Wireless Communications*, vol. 19, no. 1, 2012.

[2] K. G. Shin, H. Kim, A. W. Min, and A. Kumar, "Cognitive radios for dynamic spectrum access: from concept to reality," *IEEE Wireless Communications*, vol. 17, no. 6, 2010.

[3] J. Howe, *Crowdsourcing: How the power of the crowd is driving the future of business*. Random House, 2008.

[4] S. Roy, K. Shin, A. Ashok, M. McHenry, G. Vigil, S. Kannam, and D. Aragon, "Cityscape: A metro-area spectrum observatory," in *Computer Communication and Networks (ICCCN), 2017 26th International Conference on*, pp. 1–9, IEEE, 2017.

[5] H. Xiong, D. Zhang, L. Wang, J. P. Gibson, and J. Zhu, "Eemc: Enabling energy-efficient mobile crowdsensing with anonymous participants," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 6, no. 3, p. 39, 2015.

[6] A. Nika, Z. Zhang, X. Zhou, B. Y. Zhao, and H. Zheng, "Towards commoditized real-time spectrum monitoring," in *Proceedings of the 1st ACM workshop on Hot topics in wireless*, pp. 25–30, ACM, 2014.

[7] A. Nika, Z. Li, Y. Zhu, Y. Zhu, B. Y. Zhao, X. Zhou, and H. Zheng, "Empirical validation of commodity spectrum monitoring," in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, pp. 96–108, ACM, 2016.

[8] M. Z. Zheleva, R. Chandra, A. Chowdhery, P. Garnett, A. Gupta, A. Kapoor, and M. Valerio, "Enabling a nationwide radio frequency inventory using the spectrum observatory," *IEEE Transactions on Mobile Computing*, vol. 17, no. 2, pp. 362–375, 2018.

[9] L. Shi, P. Bahl, and D. Katabi, "Beyond sensing: Multi-ghz realtime spectrum analytics.," in *NSDI*, pp. 159–172, 2015.

[10] F. Ma, X. Liu, A. Liu, M. Zhao, C. Huang, and T. Wang, "A time and location correlation incentive scheme for deep data gathering in crowdsourcing networks," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[11] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proceedings of the 18th annual international conference on Mobile computing and networking*, pp. 173–184, ACM, 2012.

[12] J.-S. Lee and B. Hoh, "Sell your experiences: a market mechanism based incentive for participatory sensing," in *2010 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 60–68, IEEE, 2010.

[13] C. H. Liu, B. Zhang, X. Su, J. Ma, W. Wang, and K. K. Leung, "Energy-aware participant selection for smartphone-enabled mobile crowd sensing," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1435–1446, 2017.

[14] C. H. Liu, J. Fan, P. Hui, J. Wu, and K. K. Leung, "Toward qoi and energy efficiency in participatory crowdsourcing," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4684–4700, 2014.

[15] W. Wang, H. Gao, C. H. Liu, and K. K. Leung, "Credible and energy-aware participant selection with limited task budget for mobile crowd sensing," *Ad Hoc Networks*, vol. 43, pp. 56–70, 2016.

[16] Y. Zhang and M. van der Schaar, "Robust reputation protocol design for online communities: A stochastic stability analysis," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 5, pp. 907–920, 2013.

[17] L. J. H. Viveros, D. A. L. Sarmiento, and N. E. V. Parra, "Modeling and prediction primary nodes in wireless networks of cognitive radio using recurrent neural networks," 2018.

[18] M. Zheleva, R. Chandra, A. Chowdhery, A. Kapoor, and P. Garnett, "Txminer: Identifying transmitters in real-world spectrum measurements," in *Dynamic Spectrum Access Networks (DySPAN), 2015 IEEE International Symposium on*, pp. 94–105, IEEE, 2015.

[19] C. Hasan and M. K. Marina, "Channel vacancy forecasting for communication-free inter-operator spectrum sharing in small cell networks,"

[20] J.-C. Yoo and T. H. Han, "Fast normalized cross-correlation," *Circuits, systems and signal processing*, vol. 28, no. 6, p. 819, 2009.

[21] H. Schreier, J.-J. Orteu, and M. A. Sutton, *Image correlation for shape, motion and deformation measurements*. Springer US, 2009.

[22] J. A. del Peral-Rosado, R. Raulefs, J. A. Lpez-Salcedo, and G. Seco-Granados, "Survey of cellular mobile radio localization methods: From 1g to 5g," *IEEE Communications Surveys Tutorials*, vol. 20, pp. 1124–1148, Secondquarter 2018.

[23] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, "CRAWDAD dataset epfl/mobility (v. 2009-02-24)." Downloaded from https://crawdad.org/epfl/mobility/20090224/cab, Feb. 2009. traceset: cab.

[24] D. T. Wagner, A. Rice, and A. R. Beresford, "Device analyzer: Understanding smartphone usage," in *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pp. 195–208, Springer, 2013.