Efficient Distribution of Sensing Queries in Public Sensing Systems

Patrick Baier, Frank Dürr, Kurt Rothermel
Institute of Parallel and Distributed Systems
Universität Stuttgart
70569 Stuttgart, Germany
Email: <patrick.baier|frank.duerr|kurt.rothermel>@ipvs.uni-stuttgart.de

***Abstract*—The advent of mobile phones paved the way for a new paradigm for gathering sensor data termed "Public Sensing" (PS). PS uses built-in sensors of mobile devices to opportunistically gather sensor data. For instance, the microphones that are available in a crowd of mobile phones can be used to capture sound samples, which can be used to construct a city noise map.**

A great challenge of PS is to reduce the energy consumption of mobile devices since otherwise users might not be willing to participate. One crucial part in the overall power consumption is the energy required for the communication between the mobile devices and the infrastructure. In particular, the communication required for sending sensing queries to mobile devices has been largely neglected in the related work so far.

Therefore, in this paper, we address the problem of minimizing communication costs for the distribution of sensing queries. While existing systems simply broadcast sensing queries to all devices, we use a selective strategy by addressing only a subset of devices. In order not to negatively affect the quality of sensing w.r.t. completeness, this subset is carefully chosen based on a probabilistic sensing model that defines the probability of mobile devices to successfully perform a given sensing query.

Our evaluations show that with our optimized sensing query distribution, the energy consumption can be reduced by more than 70% without significantly reducing the quality of sensing.

***Keywords*-Mobile Computing, Energy-aware systems**

I. Introduction

Modern mobile phones are equipped with various sensors such as positioning sensors (GPS), accelerometers, magnetic field sensors, light sensors, acoustic sensors (microphone), or barometers. Moreover, fast communication technologies like 3G networks or WiFi together with cheap flat rates are available, which has led to an "always on" usage pattern where mobile phones are constantly connected to the Internet and can access server infrastructures.

These technology trends effectively turn mobile phones into powerful networked sensor platforms, which can be utilized to gather environmental data. Using mobile phones for sensing has been termed "Public Sensing", "Mobile Phone Sensing", "Urban Sensing", or "Participatory Sensing" in the literature (throughout this paper, we use the term "Public Sensing" (PS)). The fact that PS can be used on-demand to gather data of larger geographic areas without big upfront investments makes it an attractive alternative to classical sensor networks, in particular, in densely populated areas such as urban centers.

Despite these advantages, implementing a PS system also raises new challenges. In particular, minimizing the energy consumption of mobile phones is a prerequisite since otherwise users might not be willing to participate in PS. Basically, energy is consumed by two operations, namely, sensing data including positioning to geo-reference the sensed data, and communication including the transmission of sensed data as well as the messages for tasking mobile devices to sense [1]. Although a number of approaches for optimizing sensing have been proposed, including our own work [2], [3], only few approaches tackle the problem of optimizing the communication part of energy consumption. However, as for instance shown by Balasubramanian in [4], mobile communication consumes a considerable amount of energy. Hence, optimizing communication is essential for the implementation of an energy-efficient PS system.

Therefore, in this paper, we focus on optimizing the communication part of PS, more precisely, the distribution of sensing queries. Most existing PS systems perform tasking of mobile devices by simply broadcasting a sensing query to all available devices. Obviously, this simple strategy wastes energy for devices that cannot participate in sensing, for instance, due to their geographic distance to the sensing query. Consider, for instance, a sensing query for sensing the temperature and noise level at the Big Ben in London within the next ten minutes. Broadcasting this query to all mobile devices in London certainly involves many devices which cannot reach Big Ben within the given deadline.

The main contribution of this paper is therefore an *efficient sensing query distribution mechanism* that only sends queries to devices that are likely to actually execute the given sensing query successfully. Since we assume that we cannot control the movement of devices in our PS system—i.e., sensing is performed opportunistically, whenever a device is close to the location of a sensing query—, we cannot be perfectly sure that a device actually comes close enough to execute a given query before the sensing deadline. Therefore, we present the design of a *probabilistic sensing model*. This model describes the likelihood that a mobile device is able to sense data for a given sensing query by utilizing spatial information about the device provided by a specifically designed location update protocol. Based on this model, we present an algorithm for selecting a minimum set of devices to receive a given query such that the chances of successful sensing this query are not negatively affected. In our evaluations we show that this optimization reduces the energy consumption by more than 70% compared to an implementation that distributes a given sensing query to all available mobile devices.

The rest of this paper is structured as follows. After giving an overview of the related work in Section II, we introduce the underlying system model of our approach in Section III. In Section IV and V, we introduce the query model of our PS system and the problem statement. The basic PS system is then introduced in Section VI. The main contributions of this paper are our probabilistic sensing model and the device selection algorithm for query distribution, which are presented in Section VII and VIII, respectively. Finally, in Section IX we evaluate the performance of our approach before Section X concludes this work.

II. Related Work

Most of the work which has been published in the field of PS so far presents architectural frameworks that do not deal with algorithmic details of saving energy. A good overview of these systems is given in the survey of Lane et al. [5].

Only recently, energy efficiency also got into the focus of PS research. The approaches for saving energy can be classified into concepts for reducing the number of sensing operations and concepts to reduce communication. In order to reduce sensing costs, we investigated techniques to reduce the energy required for positioning in previous work [2], [6]. Moreover, in [3] and [7] we presented cooperative sensing algorithms that coordinate sensing between mobile devices to avoid redundant sensor readings. A more generic way to increase energy efficiency has been proposed by Priyantha et al. [8] and Lu et al. [9], where the former proposed dedicated hardware components to enable energy-efficient sensing. The latter introduced sensor-specific pipelines to reduce the processing costs of recorded sensor data.

Considering the aspect of efficient query distribution, only a few approaches have been proposed so far. Lu et al. [10] showed with their "Bubble-Sense" system how a sensing query can be "pinned" to a physical location by periodically exchanging ad-hoc messages between the devices near that location. However, this concept relies on the wide availability of ad-hoc communication interfaces on all devices, which are still hard to configure and therefore are rarely available in practice. Moreover, if no device is nearby the location of the sensing query, the query is lost and can only be restored by a node that has stored the query.

Other concepts that are more closely related to our work were proposed by Reddy et al. [11], Ruan et al. [12] and Cardone et al. [13]. Similar to our work, their approaches choose a subset of mobile devices to which a sensing query

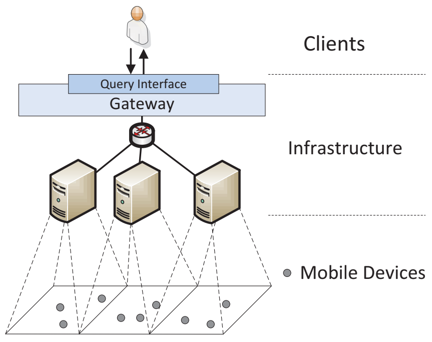


Figure 1. System Architecture

is distributed. To decide which devices are most suitable for being included into that subset, their approaches analyze historic information about the user movement patterns and extract the places were the user regularly resides. However, this requires that the user agrees to store a privacy critical user profile of its movement history.

Therefore, the query distribution approach we present in this paper uses a different strategy. First, we do not have to analyze the user profile but rather estimate the future movement of a user based only on its last known position. Hence, obfuscation techniques can be used on top of our approach to hide individual user IDs. Moreover, we are the first so far to provide a probabilistic sensing model that is used for selecting the appropriate devices for query distribution to account for the inevitable uncertainty of opportunistic sensing.

III. System Model

Next, we present the architecture of our PS system (see Fig. 1). Our system consists of a computing infrastructure and a set of mobile devices. Mobile devices are carried by their owners, who are moving according to an underlying road graph. Since we strive for an opportunistic sensing approach, we assume that the movement of the mobile devices cannot be influenced by the system. Each device is equipped with a set of sensors for recording environmental phenomena, a positioning system (e.g., GPS), and a mobile communication interface such as UMTS or LTE.

The computing infrastructure of the system consists of a set of servers and a central gateway that serves as query interface for clients that request sensor data from the system. For reasons of scalability, the geographical coverage of the sensing system is partitioned into disjoint service areas, where each of the servers is responsible for one service area. A server can communicate with the mobile devices located in its service area via mobile communication. Furthermore, we assume that a server is aware of all mobile devices that are currently located in its service area. This can be implemented by a node registration mechanism like the one used in cellular networks, for instance. The servers are connected with each other and with the gateway through a fixed communication network.

To characterize the energy consumption of the mobile devices for mobile communication, we follow the measurements of Balasubramanian et al. [4]. They showed that the energy for transferring small pieces of data is negligibly small in comparison to the energy that is needed for powering up and down the communication interface. More details of this energy model are presented in Section IX.

IV. Query Model

In order to start the gathering of sensor data for a certain location, a client sends a sensing query to the gateway. In the following, we explain the query parameters in more detail.

A sensing query contains the following parameters:

- The sensing range, which is defined by the center coordinates (x, y) and the radius r_q.
- The sensor type s_t, which defines the type of the requested sensor data.
- The sensing deadline t_d, which defines the time until when the sensor data must be recorded.

A sensing query is distributed to a subset of mobile devices, as presented later in detail, which try to record sensor data for that query using their build-in sensors. A sensing query q, which was issued at time t_q, is satisfied if there is at least one mobile device m that fulfills the following two conditions:

1) $\exists t \in [t_q, t_d] : d((x, y), pos_m(t)) \leq r_q$, where $d(\cdot, \cdot)$ denotes the Euclidean distance and $pos_m(t)$ the position of m at time t.
2) $s_t \in m.types$, where $m.types$ denotes the sensors that are available on device m.

To simplify matters, we neglect the inaccuracy of the positioning system in the description of our concepts. Nevertheless, all concepts can be extended to take the positioning error into account using techniques like map-matching. Furthermore, we limit our considerations to the case in which data for only a certain sensor type is queried and denote the set of all devices that are equipped with this sensor as M. For cases in which different types of sensor data are queried, the following concepts can be extended by defining an individual set of devices for each sensor type.

V. Problem Statement

Simply broadcasting a sensing query to all mobile devices $m \in M$ would cause a high energy consumption on the devices, since every received message consumes additional energy. In particular, this is critical if many sensing queries are issued to the PS system. The problem of this naive distribution is that it includes also devices that cannot participate in sensing until the query deadline because of their spatial distance to the sensing range.

In our optimized approach, we tackle this problem by distributing the query to only a subset of devices (called the *recipient set* $R \subseteq M$) that have a high probability of actually sensing the query successfully. However, reducing the recipient set also increases the risk that a sensing query is missed, since not all devices are aware of the query. Our goal is now to find a minimal recipient set that achieves the same sensing result as the naive distribution.

Let $\sigma(M) \in \{true, false\}$ denote the successful or unsuccessful execution of a given query q when q is distributed to all $m \in M$. Our problem can now be formalized as:

$$\begin{aligned} \underset{R \subseteq M}{\text{minimize}} \quad & |R| \\ \text{subject to} \quad & \sigma(R) = \sigma(M) \end{aligned} \tag{1}$$

Sending q only to R instead of M avoids the energy for receiving q on all devices in $M \backslash R$. This is in particular very beneficial if M is rather large or if many queries are distributed to the mobile devices.

Note that the size of the minimal R is 1 if $\sigma(M) = true$ and 0 if $\sigma(M) = false$. However, since we cannot foresee the value of $\sigma(M)$ at decision time and do not know the future movement of the mobile devices, in most cases we will choose a larger R to ensure that the given sensing constraint is fulfilled. In particular, we will include all mobile devices in R that are very likely to sense q before the sensing deadline. For this purpose, we present in the following a probabilistic sensing model that predicts the sensing probability for each device. Based on this probabilistic model, we present a recipient selection algorithm that chooses a recipient set that promises high sensing success.

VI. Basic Sensing System

To begin with, we first present a basic sensing systems that implements a simple query distribution serving as a reference for evaluating the energy efficiency of our optimized approach. To illustrate the operations of this system, we describe its processing steps with the help of a sensing query.

Assume a client queries the system by sending a query q to the query interface of the gateway. The gateway forwards q to the server whose service area intersects with the sensing range of q. If the sensing range overlaps with the service areas of multiple servers, the query is forwarded to all of these servers, and the server covering the center (x, y) of query q operates as coordinator. For the rest of this paper, we only consider the case where q is forwarded to one server and do not consider the coordination in more detail.

The query distribution algorithm that runs on the server is illustrated in Fig. 2. As indicated in lines 2–4, q is distributed to every mobile device $m \in M$. A mobile device that received a sensing query constantly checks if it is in the sensing range of the received query. In this case, it records data for q and uploads the recorded data to the server. As soon as the server receives sensor data from a mobile device m_r, it returns the sensed data back to the gateway, which forwards the query result to the client. To avoid energy

```
1:  procedure QUERY DISTRIBUTION($q$)
2:      $D \leftarrow M$        ▷ Modified in optimized approach
3:      for all $m \in D$ do
4:          SENDQUERY($m, q$)
5:      $result \leftarrow 0$
6:      while $(t_{now} < t_d) \wedge (result = 0)$ do
7:          $result \leftarrow$ RECEIVERESULT($m_r$)
8:      if $result = 0$ then
9:          return $error$
10:     for all $m \in D \backslash \{m_r\}$ do
11:         STOPQUERYMSG($m, q$)
12:     return $result$
```

Figure 2. Query Distribution Algorithm

consuming redundant sensor readings on the mobile devices, the server subsequently sends a message to all devices that initially received q to stop the execution of the query (lines 10–11). If q is not sensed until the sensing deadline t_d, the sensing query fails, and the server returns an error message to the gateway, which in turn informs the client (lines 8–9).

VII. Probabilistic Sensing Model

The basic idea of our optimized approach is to reduce the query distribution to only a subset $R \subseteq M$ (line 2 in Fig. 2 is replaced by $D \leftarrow R$) and include only those devices in R that promise a high sensing success. To this end, we present a probabilistic sensing model defining the probability that a device satisfies a given sensing query.

We introduce the probabilistic sensing model in four steps: First we introduce the concept of location updates, which is needed to obtain spatial information about the devices on the server. Then, we present a movement prediction model that estimates future device positions based on the last location update. Subsequently, we define the sensing probability for a device on a single path and then extend this definition to reflect the sensing probability over multiple paths.

A. Location Update Protocol

In the basic sensing system, a server knows which devices are currently located in its service area but has no information about their spatial distribution within this area. However, to determine the probability that a device can satisfy a given sensing query, an estimation about the current device position is needed. To provide this information, we require a device to run a location update protocol which triggers the device to send location update messages to the server.

Since sending the device position to the server consumes additional energy and is therefore critical for our goal of increasing energy efficiency, we have designed a specific location update protocol that adapts to the needs of the query distribution scenario. The problem of existing location update protocols (for a comparison see [14], [15]) is that they trigger a mobile device to update its position on the server in a proactive manner. For instance, the distance-based update protocol triggers a location update message whenever the current position of the mobile device deviates by more than a predefined threshold from the position on the server. However, if sensing queries are issued only very rarely, the overhead for using such an update protocol can exceed the energy savings that can be achieved using our adaptive query distribution, which requires location information about the devices. As a result, these proactive update protocols are not suitable for our scenario.

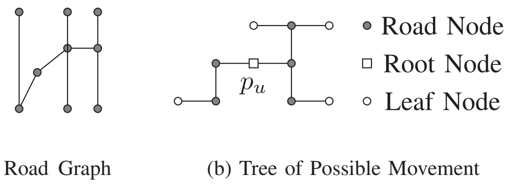
Instead, we propose a passive location update protocol that adapts to the number of sensing queries and increases energy efficiency by sending location updates in an opportunistic fashion. As mentioned in Section III, the energy overhead for powering up and down the communication interface is much larger than the actual transmission energy for small messages. We utilize this characteristic for our update protocol as follows: A device updates its current position on the server after each communication with the server (line 4, 7 and 11 in Fig. 2). Since the communication interface of the device is already powered up, the energy for sending a location update messages is rather small. The second advantage of this protocol is that the number of location updates scale with the number of sensing queries that are issued to the system. As we will see later on, the overhead of this update protocol will not exceed the energy savings from our approach even if only few queries need to be distributed.

Note that if only a small number of location updates will be sent, the device position on the server can get very inaccurate over time. However, we will see that our approach also works on inaccurate position information and automatically adapts the size of recipient set R to account for this case. On the other hand, the frequency of location updates is limited by the sampling time of the positioning sensor, i.e., updates are only sent when the positioning sensor provides a new position. In the following, we refer to the position of the device that is stored on the server as p_u.

B. Movement Prediction Model

For inferring sensing probabilities, it is important to know whether the sensing range of a given sensing query lies on the future movement path of a certain mobile device. For this purpose, the movement prediction model enumerates the possible movement paths of a device and assigns each path a probability that the device will actually take this path.

As mentioned earlier, mobile devices move on a road graph that is known to the server. This road graph is defined by a set of road nodes and a set of edges that interconnect these nodes (see Fig. 3a). For each road node r, the set of its adjacent road nodes is given as A_r. To reflect the significance of different roads, an individual turning function



(a) Road Graph Geometry (b) Tree of Possible Movement

Figure 3. Road Graph and Movement Tree

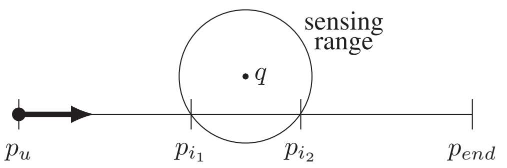


Figure 4. Line of Movement

$f_r : A_r \times A_r \rightarrow [0,1]$ is given for each road node. The value of $f_r(r_a, r_b)$ depicts the probability that a mobile device leaves the road node r towards road node r_b when having entered it via road node r_a. This function can for example be derived from road traffic analysis [16].

For predicting the movement of device m, we enumerate the different paths that the device can take starting from its last known position p_u. However, since the number of possible paths grows exponentially over time, we limit this number by assuming that m moves towards some unknown destination on the graph traversing the shortest path (which is also the basic assumption of most sophisticated mobility models, for instance the *random waypoint model*). The movement of m from time t_u (the time of the last location update p_u) until sensing deadline t_d can then be described by a tree of road nodes which is rooted at p_u (see Fig. 3b). The leaves of this tree consists of all points on the road graph that are distance $s_{max} = (t_d - t_u) \cdot v_{max}$ away, where v_{max} depicts the maximum speed of device m. Note that the value of v_{max} can either be obtained directly from the device or some value on the device speed can be assumed (for instance, by taking speed limits of the road graph into account). Each path from the root node towards a leaf describes one possible movement path w that m can take until t_d. We describe a path w by the nodes that are traversed on this path, i.e., $w = (r_{root}, \cdots, r_{leaf})$.

To reflect the probability that a devices takes a certain path, we introduce the random variable X_w. The probability $P(X_w = w)$ that m takes path w is then given by multiplying the respective turning probabilities at the encountered road nodes when traversing w:

$$P(X_w = w) = \prod_{i=1}^{i<|w|-1} f_{w[i]}(w[i-1], w[i+1]) \qquad (2)$$

Having defined the probability that a device moves on a given path, we will define in the following the probability that it can sense a given query under the condition that it moves on this path.

C. Single Path Sensing Probability

To denote the sensing probability we introduce the random variable $X_s \in \{0,1\}$, which reflects the case that a device m can sense a given query q ($X_s = 1$) or not ($X_s = 0$). We will first derive the distribution of X_s under the assumption that m takes a certain path w, i.e., $P(X_s \mid X_w = w)$, before we derive the general distribution of X_s in the next subsection.

Assuming that m traverses path w, the line segment $[p_{i_1}, p_{i_2}]$ that intersects w with the sensing range of q contains the points from which m can sense q (see Fig. 4). In the following, we refer to the end of path w, which is s_{max} away from p_u, as p_{end}. To determine whether m can sense q, we are interested in the likelihood that m passes line segment $[p_{i_1}, p_{i_2}]$ in the time frame between q is queried (referred to as t_q) and sensing deadline t_d. To this end, we derive in the following a probabilistic representation of the position of device m for time t_q and t_d.

To consider a device's movement speed, we assume that a probability density function (pdf) $f_{X_v}(v)$ of the random variable X_v is given, which describes the average speed of the device for an arbitrary time interval. We assume that the values of X_v for two disjoint time intervals are independent. This pdf can either be provided directly by the device or we can just assume an appropriate probability distribution to approximate the device speed (e.g., Gaussian distribution). The distance d that a device passes in a given time period Δt is then given by the following formula:

$$d = X_v \cdot \Delta t$$

We denote the distance that m passes on w between time t_u, when the last location update was received, and t_q as X_{d_1}. Analogous, we denote the distance that m passes between t_q and t_d as X_{d_2}. Both are defined as follows:

$$X_{d_1} = X_v \cdot (t_q - t_u)$$
$$X_{d_2} = X_v \cdot (t_d - t_q)$$

The two variables X_{d_1} and X_{d_2} are a linear transformation of the random variable X_v. Therefore, they are both random variables as well. The pdfs for the linearly transformed variables are then given as follows:

$$f_{X_{d_1}}(d_1) = \frac{1}{t_q - t_u} \cdot f_{X_v}\left(\frac{d_1}{t_q - t_u}\right)$$
$$f_{X_{d_2}}(d_2) = \frac{1}{t_d - t_q} \cdot f_{X_v}\left(\frac{d_2}{t_d - t_q}\right)$$

Based on these probabilistic definitions, the position of m on w at time t_q is given as $p_u + X_{d_1}$ and as $p_u + X_{d_1} + X_{d_2}$ for time t_d. We can now formulate the case in which m passes line segment $[p_{i_1}, p_{i_2}]$ in the time frame $[t_q, t_d]$ by the following two conditions:

1) When m receives q from the server at time t_q, it must not have moved further than point p_{i_2} on w. Otherwise m has already passed q when it receives the query.
2) When the sensing deadline of q is reached at time t_d, m must have reached at least point p_{i_1} on w.

More formally, we say a device m can sense q on a given path if the following two conditions are fulfilled:

$$X_{d_1} \leq \overline{p_{i_2}} \tag{3}$$

$$X_{d_1} + X_{d_2} \geq \overline{p_{i_1}} \tag{4}$$

Here, $\overline{p_i}$ depicts the distance between p_u and p_i on path w. Note that due to our shortest path assumption, we can assume that a device only moves towards the end of path w.

To formulate these two conditions in a closed form, we need to define the joint pdf of the random variables X_{d_1} and $X_{d_1} + X_{d_2}$. Due to the independence of X_{d_1} and X_{d_2}, we get the convolution of the pdf of both variables as:

$$\begin{aligned} f_{X_{d_1}, X_{d_1}+X_{d_2}}(d_1, d) &= f_{X_{d_1}, X_{d_2}}(d_1, d - d_1) \\ &= f_{X_{d_1}}(d_1) \cdot f_{X_{d_2}}(d - d_1) \end{aligned}$$

We can now formulate the probability that m senses q when moving on path w by integrating this joint pdf over the boundaries that we defined in Equation 3 and 4:

$$\begin{aligned} & P(X_s = 1 | X_w = w) \\ &= P(X_{d_1} \leq p_{i_2}, p_{i_1} \leq X_{d_1} + X_{d_2} \leq p_{end}) \\ &= \int_0^{\overline{p_{i_2}}} \int_{\overline{p_{i_1}}}^{\overline{p_{end}}} f_{X_{d_1}}(x) \cdot f_{X_{d_2}}(y - x) \, dy dx \end{aligned} \tag{5}$$

Note that we assume that there are only two intersection points between w and the range of q. Cases in which the sensing range intersects the path on more than two points can be handled by summing up the single probabilities for each intersection using the inclusion-exclusion method [17].

D. Multiple Path Sensing Probability

To get the overall probability $P(X_s = 1)$ that a device can sense a given query q, we have to combine the definitions from Equations 2 and 5. Given the probability that device m moves on path w and the probability that m can sense q if it moves on w, we apply Bayes' Theorem as follows:

$$P(X_s = 1) = \sum_{\forall w_i} P(X_s = 1 \mid w_i) \cdot P(X_w = w_i) \tag{6}$$

In the following, we denote the probability $P(X_s = 1)$ for device m sensing query q as $p_{m,q}$.

VIII. Recipient Selection

We can now use these definitions to find a recipient set for the query distribution. The basic idea is to define the joint sensing probability for a given set of devices and use this metric to find a recipient set that promises high energy efficiency and sensing success.

The joint sensing probability for a given set of devices D is given as the probability that query q will be satisfied when it is distributed to all $m \in D$. To this end, we use the single device sensing probability from Equation 6 and calculate the joint sensing probability p_q for D as:

$$p_q(D) = 1 - \prod_{\forall m \in D} (1 - p_{m,q}) \tag{7}$$

This function can be derived from the special case of a binomial distribution, where at least one event is true. Note that we can easily extend our system by requiring that a sensing query is only satisfied if it is sensed by k different devices. In this case, Equation 7 has to be adapted to consider k sensing events (see [17] for the combinatoric details).

To find a promising recipient set R, we use this definition to quantify the sensing success of different $R \in 2^M$ in order to find a small R that has a high sensing probability. As a first step, we limit the solution space of this decision by restricting the number of devices that are candidates for being included in R. As we can conclude from Equation 7, only those devices are valuable candidates that have a sensing probability greater than zero. Hence, we define the *candidate set*, which is given as $C = \{m \in M \mid p_{m,q} > 0\}$. Now, we define R by greedily including the mobile device with the highest sensing probability in C until $p_q(R)$ reaches a predefined system parameter $\mu_p \in (0, 1]$, which we refer to as the *target probability*. In the evaluation, we will see which values of μ_p will result in a high energy efficiency.

In more detail, R is computed by the following algorithm:

```
1:  procedure RECIPIENT SET SELECTION($C, \mu_p$)
2:      $R \leftarrow \emptyset$
3:      $p_q \leftarrow 1$
4:      repeat
5:          $m \leftarrow$ GETMAXPROBDEVICE($C$)
6:          $R \leftarrow R \cup \{m\}$
7:          $C \leftarrow C \setminus \{m\}$
8:          $p_q \leftarrow p_q \cdot (1 - p_{m,q})$
9:      until $(1 - p_q \geq \mu_p) \vee (R = C)$
10:     return $R$
```

The function call in line 5 returns the device with the currently largest value of $p_{m,q}$ in C. The algorithm stops when the joint sensing probability of R reaches the target probability μ_p or when R is equivalent to C. In the latter case, the target probability μ_p cannot be reached. However, to provide a best effort solution by maximizing the sensing probability, the algorithm returns R, which is equal to C.

Note that this algorithm automatically adapts the size of the recipient set to the accuracy of the device positions on the server. More precisely, if a device has not updated its position on the server for a long time, the probability $p_{m,q}$ is rather small, since the device may have taken a large number of possible paths. As a result, many devices have to be included in R to reach target probability μ_p .

IX. Evaluation

In this section, we present the evaluation results of our system. Since real-world experiments would require a large number of physical devices and participants, we evaluated our system with the help of a mobility simulator. Before we present the simulation results, we shortly introduce the simulation setup and the energy model we used.

A. Simulation Setup

We used the ONE network simulator [18] to implement the concepts presented in this paper. To allow for the calculation of sensing probabilities for several hundreds of devices in real-time, we precalculated the shortest path distances for each pair of road nodes in advance. Furthermore, we approximated the integrals for the calculation of the probabilistic sensing model numerically. For simulating the device mobility, we used the built-in mobility traces files of the simulator, which implement a random shortest path movement of pedestrians on the road graph of Helsinki with size $15\,\mathrm{km}^2$. We assume not to have any prior knowledge about the turning probabilities at intersections on this map and, thus, assume them to be uniformly distributed. Having accurate information about turning probabilities would even improve our concepts, hence, the following results can be seen as lower bound on the performance of our approach. We simulated 3 hours of pedestrian movement and generated a sensing query per minute at a random position on the map with a sensing deadline of 300 seconds.

B. Energy Model

To characterize the energy of the mobile device, we use the widely referenced model of Balasubramanian et al. [4]. They subdivide the energy consumed by the mobile devices for transmitting data into three parts. The *ramp energy* is needed for powering up the communication interface and is given as $3.5\,\mathrm{J}$. The *transmission energy* is consumed for the data transmission and is given as $0.025\,\mathrm{J/KB}$. The *tail energy* is the energy that the interface consumes until it is powered down. The tail time is given as $12.5\,\mathrm{s}$ and the interface consumes $0.62\,\mathrm{J/s}$ until it powers down. This leads to the following energy model that describes the energy consumption of receiving $x\,\mathrm{KB}$ data at time t when the last data transfer ended at time t':

$$E = \begin{cases} 0.025x + (t - t') \cdot 0.62\,\frac{J}{s} & \text{if } t - t' \leq 12.5\,s \\ 0.025x + 12.5\,s \cdot 0.62\,\frac{J}{s} + 3.5\,J & \text{else} \end{cases}$$

Balasubramanian et al. have also shown that the energy for sending and receiving is equivalent for message sizes below 1 KB. Hence, we use the above energy model for both cases.

For the rest of this paper, we only consider the communication energy (i.e., for receiving sensing queries and sending location updates). Also considering the energy that is consumed by the devices for positioning (e.g., GPS) would add the same amount of energy in all approaches and is therefore not decisive.

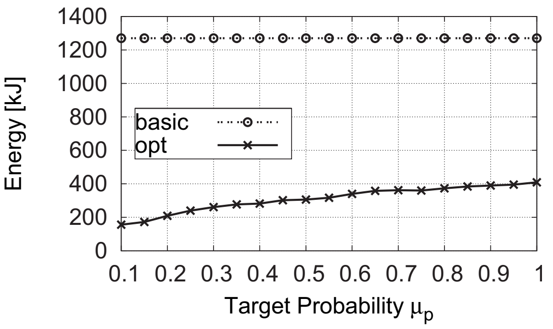


Figure 5. Energy Consumption against μ_p

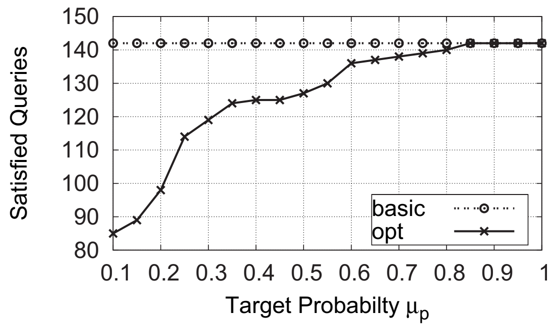


Figure 6. Query Success Rate against μ_p

C. Energy Efficiency and Sensing Success

To begin with, we look at the energy efficiency and the sensing success of the query distribution.

Fig. 5 shows the total energy consumption cumulated over all mobile devices for different simulation scenarios. In each scenario, 500 mobile devices were simulated and a different value for the target probability parameter μ_p (see Section VIII) was chosen. The figure shows the results of the basic approach and the optimized query distribution. First, we see that the basic approach consumes significantly more energy in every scenario than the optimized approach. Even if μ_p is set to 1, i.e., a query is distributed to all devices in the candidate set, the optimized approach consumes $68\,\%$ less energy than the basic approach. For smaller values of μ_p, the energy consumption can be reduced even more.

To see if this energy reduction has a negative impact on the sensing success, we compare the number of successfully satisfied queries for the same scenarios as before (see Fig. 6). For this analysis, the basic approach serves as a base line, since it achieves the best possible sensing result by distributing a sensing query to all devices. The figure shows that setting the target probability to higher values than 0.8 results in the same number of satisfied queries as in the basic

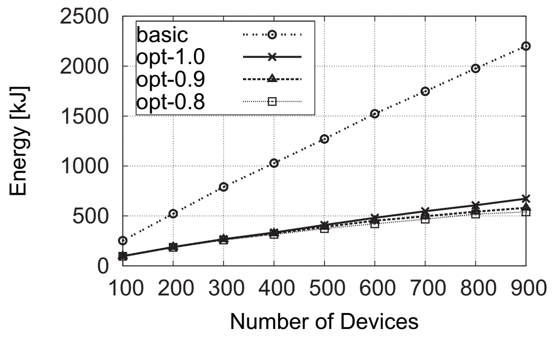


Figure 7. Energy Consumption against # Devices

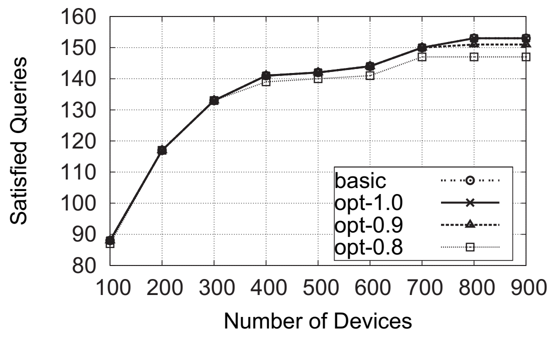


Figure 8. Query Sucess Rate against # Devices

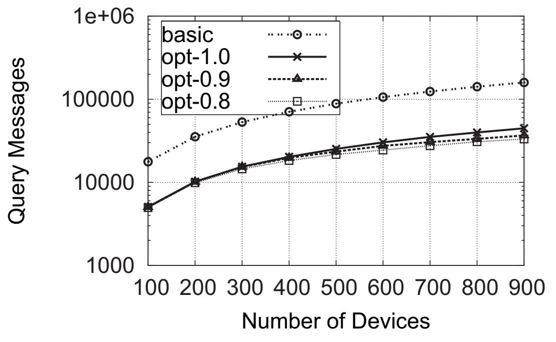


Figure 9. Message Overhead for Query Distribution

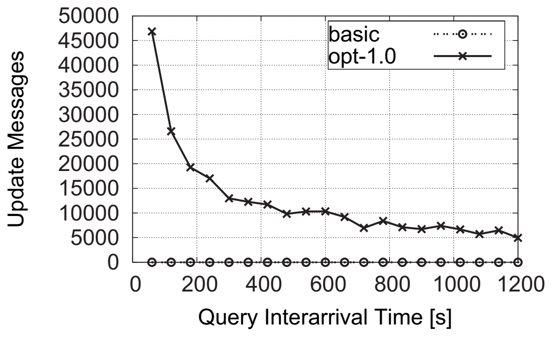


Figure 10. Message Overhead for Location Updates

approach. Even if μ_p is set to 0.6, still $95\,\%$ of the queries are satisfied compared to the basic approach.

These results show that the energy consumption for query distribution can be reduced significantly without reducing the sensing success. In particular, setting the target probability parameter to $\mu_p = 0.85$, our approach reduces the energy consumption by $70\,\%$ compared to the basic approach while the same number of queries are satisfied.

To see how energy and sensing success vary over device density, Fig. 7 and 8 show the same results for different number of mobile devices in the simulation. We see that the energy savings increase with the number of devices. This clearly highlights the benefit of our approach: While the energy for query distribution grows linearly with the number of devices in the basic approach, our approach limits the number of query receivers in order to save energy. Furthermore, choosing $\mu_p = 0.8$, the average energy consumption over all scenarios can be reduced by $70\,\%$ while on average $98\,\%$ of the queries can be satsified compared to the basic approach. From this analysis we can conclude that $\mu_p = 0.8$ is a good parameter setting in order to reduce energy without reducing sensing success in all scenarios.

D. Message Overhead

To understand how the presented energy values are composed, we look at the number of messages that are sent in the system. Fig. 9 and 10 show the message overhead for the query distribution and the location update messages, respectively. We see that the basic approach sends a large number of queries on the one hand. On the other hand, it does not require any location update messages. However, the energy spent for sending queries dominates the saved energy for location updates compared to the optimized approach. For the optimized approaches, we see that a smaller value of μ_p results in less query messages. Clearly, the smaller μ_p, the less devices are necessary to reach the target probability. Hence, less query messages are distributed.

Fig. 10 shows the number of location update messages that are sent for different scenarios in which the number of queries requested by the clients vary. To this end, the interarrival time between two consecutive queries is varied for different simulations (shown on the x-axis). We see that the number of location updates decreases with the number of queries that are issued to the system. As intended, the message overhead of our location update protocol adapts to the query distribution, which makes our approach also efficient in scenarios where sensing queries are issued only rarely, as we will see in the following.

E. Query Rate

Finally, we look at the efficiency of our system for different numbers of queries requested by the clients in Fig. 11. We see that our optimized approach performs better in relation to the basic approach, the more frequently queries are issued. On the other hand, the larger the time between two queries, the less are the energy saving of the optimized approach.

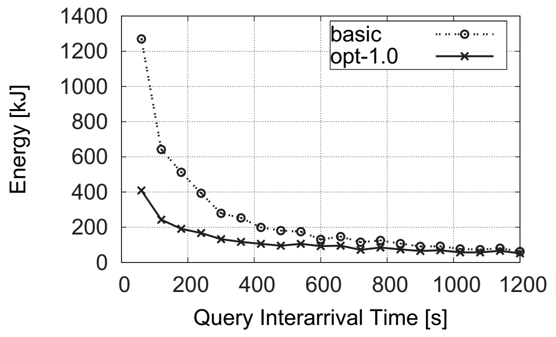


Figure 11. Energy Consumption against Query Rate

Obviously, the more queries need to be distributed, the more energy can be saved in the query distribution.

Looking at the situation when queries are requested less frequently, we see that our approach converges to the basic approach. In particular, this means that our update protocol adapts to this scenario by reducing the sending rate of location updates. Since the location information on the PS server is rather old, the recipient set selection algorithm has to include almost all devices for new queries.

X. Conclusion & Future Work

In this paper, we presented an efficient query distribution approach for PS systems. In comparison to other PS systems that simply broadcast sensing queries or rely on historical movement data of users, our approach distributes queries selectively and only relies on the last known position of a mobile device. We presented a probabilistic sensing model and an algorithm that restricts the distribution to those devices that are best suited for sensing. We showed that our approach can significantly increase the energy efficiency in comparison to a basic system, while still providing a very high sensing success rate. The total energy savings of the system increase with a higher number of mobile devices and with a higher query rate.

In future work, we are going to extend our work on query distribution in PS. Instead of considering one shot queries, we investigate the distribution of periodic sensing queries which have to be sensed in certain time intervals. To this end, we have to extend our probabilistic sensing model to consider more than one sensing of a query and introduce adaption mechanisms for the re-distribution of queries.

XI. Acknowledgements

This work was funded by the Com'N'Sense project of the German Research Foundation (DFG).

References

[1] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application," in *Proc. of the Conf. on Embedded Network Sensor Systems*, 2008.

[2] P. Baier, F. Dürr, and K. Rothermel, "Psense: Reducing energy consumption in public sensing systems," in *Proc. of the Conf. on Advanced Information Networking and Applications*, 2012.

[3] H. Weinschrott, F. Dürr, and K. Rothermel, "Streamshaper: Coordination algorithms for participatory mobile urban sensing," in *Proc. of the Conf. on Mobile Adhoc and Sensor Systems*, 2010.

[4] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy consumption in mobile phones: a measurement study and implications for network applications," in *Proc. of the Conf. on Internet Measurement*, 2009.

[5] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, "A survey of mobile phone sensing," *Communications Magazine, IEEE*, vol. 48, pp. 140–150, 2010.

[6] P. Baier, H. Weinschrott, F. Dürr, and K. Rothermel, "MapCorrect: automatic correction and validation of road maps using public sensing," in *Proc. of the Conf. on Local Computer Networks*, 2011.

[7] D. Philipp, F. Dürr, and K. Rothermel, "A sensor network abstraction for flexible public sensing systems," in *Proc. of the Conf. on Mobile Ad-Hoc and Sensor Systems*, 2011.

[8] B. Priyantha, D. Lymberopoulos, and J. Liu, "Littlerock: Enabling energy-efficient continuous sensing on mobile phones," *Pervasive Computing, IEEE*, vol. 10, pp. 12–15, 2011.

[9] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell, "The jigsaw continuous sensing engine for mobile phone applications," in *Proc. of the Conf. on Embedded Networked Sensor Systems*, 2010.

[10] H. Lu, N. D. Lane, S. B. Eisenman, and A. T. Campbell, "Bubble-sensing: Binding sensing tasks to the physical world," *Pervasive and Mobile Computing*, vol. 6, 2010.

[11] S. Reddy, D. Estrin, and M. Srivastava, "Recruitment framework for participatory sensing data collections," in *Proc. of the Conf. on Pervasive Computing*, 2010.

[12] Z. Ruan, E. C. H. Ngai, and J. Liu, "Wireless sensor deployment for collaborative sensing with mobile phones," *Comput. Netw.*, vol. 55, pp. 3224–3245, 2011.

[13] 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," *Communications Magazine, IEEE*, vol. 51, pp. –, 2013.

[14] A. Leonhardi and K. Rothermel, "A comparison of protocols for updating location information," *Cluster Computing*, vol. 4, pp. 355–367, 2001.

[15] P. Baier, F. Dürr, and K. Rothermel, "Opportunistic position update protocols for mobile devices," in *Proc. of the Conf. on Ubiquitous Computing*, 2013.

[16] M. Maher, "Estimating the turning flows at a junction: a comparison of three models," *Traffic Engineering & Control*, vol. 25, pp. 19–22, 1984.

[17] P. J. Cameron, *Combinatorics: Topics, Techniques, Algorithms*. Cambridge University Press, 1994.

[18] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in *Proc. of the Conf. on Simulation Tools and Techniques*, 2009.