Towards Planet-Scale Localization on Smartphones with a Partial Radiomap

A. Konstantinidis
Dept. of Computer Science
U. of Cyprus and Frederick U.
akonstan@cs.ucy.ac.cy

G. Chatzimilioudis
Dept. of Computer Science
University of Cyprus
gchatzim@gmail.com

C. Laoudias
KIOS Research Center
University of Cyprus
laoudias@ucy.ac.cy

S. Nicolaou
Dept. of Computer Science
University of Cyprus
snicol02@cs.ucy.ac.cy

D. Zeinalipour-Yazti
Dept. of Computer Science
University of Cyprus
dzeina@cs.ucy.ac.cy

ABSTRACT

The majority of smartphone localization systems use *Assisted-GPS* for fine-grained localization in outdoor spaces or WiFi-based *RSS (Received Signal Strength)* technologies for coarse-grain positioning in indoor and outdoor spaces. The former consumes precious energy from mobile devices, is strictly affected by the environment (e.g., cloudy day, forests, etc.) and does not work in indoor spaces. The latter collects RSS from WiFi beams within a user's vicinity and transfers an RSS vector to the server for localization, in which the position of the user is disclosed possibly violating users' privacy. In this paper, we present *BloomMap*, an innovative and efficient algorithm that conducts a localization process without unveiling the user's location to the localization service, minimizing the energy consumption of the mobile unit and also minimizing the network traffic by not transferring large positioning structures to the client (i.e., known as radiomap). Our framework is designed for planet-scale RSS localization scenarios, which are expected to emerge in the near-future. In particular, a user may localize itself using a subset of a vast data repository of RSS signals that is updated in real time by smartphone wardrivers. Our preliminary evaluation shows that our propositions can localize a device without unveiling its location in approximately 80% less time, energy and network resources than competitive approaches. We also describe our WiFi-based prototype system developed on the Android OS.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: General

Keywords

WiFi Localization, Bloom Filters, Smartphones

1. INTRODUCTION

The widespread deployment of smartphone devices has brought a revolution in location-aware applications and services for mobile phones [20]. By using a tracking mechanism available on smartphones, such as the iPhone, Blackberry and Android, users are able to post social media updates with text, pictures, and even video, tagged with specific locations in real time. For example, Google Latitude [6] enables users to track the places they and their friends have visited. Similarly, location-aware mobile social networking applications like Foursquare, Gowalla and Loopt enjoy enormous success in the Smartphone community and academic efforts in this direction are also underway [3, 10].

Currently, most localization services designed for smartphone devices rely on positioning techniques such as A-GPS and Centralized RSS (Received-Signal-Strength) Radiomap services. A-GPS localization is carried out with the assistance of beams transmitted from satellites. These beams are negatively affected from the environment (e.g., cloudy days, forests) and they do not work in indoor spaces. The Centralized RSS Radiomap localization is carried out by collecting RSS values from WiFi Access Points (APs) in the vicinity of the user and transferring an RSS vector to a centralized server that subsequently derives the user's location (e.g., Google Geolocation, Skyhook, etc.) We claim that the A-GPS services and the Centralized RSS Radiomap approaches will be severely hampered in the future due to the following constraints:

i. **A-GPS Drawback:** GPS suffers from a high-energy drain on mobile devices, thus it is expected that its utilization will be limited to spaces where other means are not available.
ii **Centralized RSS Drawback:** Continuously disclosing the RSS vector to a centralized authority means disclosing the user's location that might compromise user privacy in very serious ways.

The transmission of complete Radiomaps to the user, to avoid compromising its privacy, means continuously transferring massive amounts of planet-scale data through WiFi / 3G / 4G connections, which can deplete the precious smartphone battery faster, increase query response times and quickly degrade the network health. In this paper, we present a planet-scale data collection storage and dissemination framework that enables smartphone users to find their location without disclosing location-context meta-data to a central authority, offering energy-efficiency, high performance in terms of retrieval time and network resource conservation, simultaneously. Notice that the centralized RSS radiomap might be infinitely large as every discrete point on the planet has a different RSS vector and the given vector changes over time

Copyright is held by the author/owner(s).
HotPlanet'12, June 25, 2012, Low Wood Bay, Lake District, UK.
ACM 978-1-4503-1318-6/12/06.

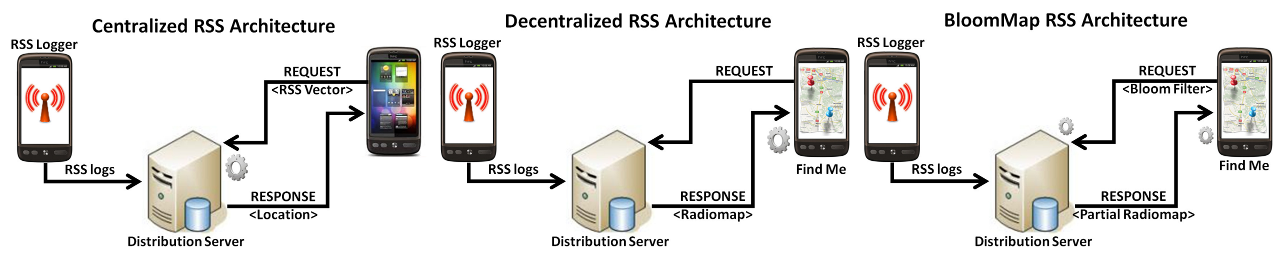


Figure 1: The architecture of the Centralized RSS (left), Decentralized RSS (center) and BloomMap (right) positioning approaches.

(thus, need to be updated by a process known as wardriving, i.e., the act of a person in a moving vehicle searching for WiFi networks using a portable computer or PDA.) In our framework, a Smartphone user runs our BloomMap algorithm that anonymously forwards a bloom filter representation of its RSS vector to a server. Bloom filters were proposed by Burton Howard Bloom, in 1970 [2], as space-efficient probabilistic data structures that are used to answer set-membership queries efficiently. The server subsequently utilizes the filter to identify the row(s) from the complete Radiomap that are necessary for the user to localize itself. The server forwards these rows to the user conserving both energy and network traffic and helps the user to localize itself without revealing its real position.

In [11], we presented our decentralized RSS localization architecture for Android smartphones, coined Airplace, which follows a mobile-based network-assisted architecture that has two highly desirable properties. A typical positioning scenario comprises of three steps: (i) a user enters an indoor environment, such as a University campus or a shopping mall, covered by several WiFi APs; (ii) the user's smartphone obtains the RSS Radiomap and parameters from the local distribution server in a single communication round; and (iii) the client positions itself independently using only local knowledge, i.e., the downloaded Radiomap and the currently observed RSS fingerprint, and more importantly without revealing its personal state. The system architecture is illustrated in Fig. 1 (center) and consists of the *RSS Logger*, the *Find Me* application and the *Distribution Server* that are detailed in Section 5.1.

We start this paper out with a description of our proposed system model and problem formulation in Section 2. In Section 3 we introduce some background information on the major concepts of the BloomMap approach followed by a presentation of the proposed algorithm that aims to minimize the smartphones' energy consumption and the retrieval time as well as to reduce the network traffic without disclosing the user's position in Section 4. In Section 5, we present our preliminary experimental study, starting with a description of our RSS prototype localization platform, coined Airplace[1] and discussing its limitations with respect to dealing with a large-scale Radiomap. Subsequently we provide the experiments in evaluating the performance of the BloomMap approach using several radiomap datasets collected locally at the University of Cyprus.

[1]Screenshots and a video presentation of our basic positioning system (without BloomMap) is available at: `http://www2.ucy.ac.cy/~laoudias/pages/platform.html`

2. PROBLEM FORMULATION

We shall now formalize the operation of the distribution server, which disseminates an RSS Radiomap to a client that seeks to localize itself. Let a server S maintain a 2-D matrix *MATRIX[N][M]*, which records the RSS value of M APs at N geo-locations *(x,y)*. For example, the Radiomap *MATRIX* can be of the following format:

```
        Radiomap (MATRIX)
AP1, AP2, .... APM => x1,y1
AP1, AP2, .... APM => x2,y2
AP1, AP2, .... APM => x3,y3
               ...
AP1, AP2, .... APM => xN,yN
```

MATRIX is typically constructed by centrally overlaying several RSS vectors:

```
AP1, AP2, .... APl => xi,yi (l<<M),
```

which are recorded by users using wardriving, i.e., the act of searching for Wi-Fi wireless networks by a person in a moving vehicle, using a portable computer or PDA. Additionally, *MATRIX* is extremely large in respect to *N*, as the M dimension is usually small and can be represented efficiently with adjacency-matrix structures. For ease of exposition, let *MATRIX* be denoted as a 2-D matrix where most points are null, e.g., *NaN* (i.e., a sparse matrix).

Current techniques conduct fine-grained positioning using a number of techniques executed on the server (e.g., Skyhook, Google, etc). For instance, Google's Geolocation API behind Google's web and mobile positioning features allows a user to ship the user's RSS vector (“$AP1, AP2, \cdots APl$”, $l << M$) to a server, where the server derive (or approximate) the *(x,y)* coordinates of the user using its *MATRIX* structure. This can be realized as follows:

Centralized Radiomap Algorithm (CRA): One could fill an XML (JSON) request with as much info as available from the user's current location, ship it over to the server and retrieve back the location of the user as accurately as possible. In this case, a user ships the MAC address and RSS value of the cell towers in its vicinity having the following characteristics:

- **Energy Consumption (Good):** CRA is an energy efficient approach, since the server performs all calculations conserving the mobile user's battery energy.
- **Retrieval Time (Good):** The processing power of the server is much higher than the user's device making the process faster. Moreover, by transmitting just the result to the user consumes minor network resources.

- **Privacy (Bad):** Disclosing the APs (i.e., RSS vector) to the server, as an input to the localization process, means disclosing coarsely the user's position.

In order to tackle the privacy concern of the CRA technique the following alternative approach may be developed:

Distributed Radiomap Algorithm (DRA): Ship the MATRIX to the client and have the client perform the mapping using one known algorithm e.g., KNN, WKNN, MAP, RBF, SNAP, etc., which will be explained in Section 5.1, having the following characteristics:

- **Energy Consumption (Bad):** The MATRIX is huge for both receiving as well as searching and finding the coordinates. This is not an efficient approach from a smartphone user's point of view.
- **Retrieval Time (Bad):** The transmission of a huge MATRIX is time consuming and also consumes more network resources.
- **Privacy (Good):** The user's position is not disclosed since the server knows nothing about the users' RSS vector.

Although the DRA approach, which is currently the underlying technique in our basic Airplace architecture, improves the data-disclosure drawback of the CRA approach, it is quite inefficient in terms of energy consumption and retrieval time in planet-scale localization scenarios. The major goal of the proposed approach is to keep the RSS vector *in-situ* for data-disclosure, offering at the same time high performance.

3. PRELIMINARIES

Before we present our *BloomMap Algorithm (BMA)*, which combines the advantages of the CRA and DRA, both of which are underlying elements of our localization proposition, we provide background knowledge about cloaking used in location privacy and Bloom filters.

3.1 Location privacy techniques

Privacy-preserving techniques for location services are based on one of the following concepts: (i) dummy locations; (ii) spatial cloaking; and (iii) space transformations. When using dummy locations the user protects their location privacy by reporting a set of fake locations termed dummies [9,18]. In (ii), the locations of users are transformed into another space in which their exact [5,17] or approximate [8] spatial relationships are maintained. The main idea in concept (iii) is to blur a user's exact location into a cloaked area that satisfies the user's privacy requirements [4,7].

As for user identification privacy, k-anonymity guarantees that the querying user is indistinguishable from at least k-1 others [15,16]. In user location privacy k-spatial anonymity is achieved by obfuscating the location of a querying user so that it cannot be identified with a probability higher than 1/k. This can straightforwardly be achieved using k dummy locations, assuming that the locations of a user has uniform probability over the space.

We will use dummy locations to achieve k-spatial anonymity and we will also cloak the user's location by sending only partial information to the server.

3.2 Bloom filters

Bloom filters were proposed by Burton Howard Bloom in 1970 [2], as space-efficient probabilistic data structures that are used to answer set-membership queries efficiently. The idea is to allocate a vector of b bits, initially all set to 0, and use h independent hash functions to hash an element to h positions in the vector with a uniform random distribution.

To create a Bloom filter for an element we feed the element to each of the h hash functions to get h vector positions and set them to 1. To test whether an element is a member of a set, we need to compare the vector of the query to the vector of the set, i.e., the Bloom filter vector created by hashing all set elements into the vector. If a single non-zero position in the query vector does not match to a non-zero position in the set vector, then the query element certainly does not exist in the set. If all non-zero positions match, then the element might be a member of the set, since Bloom filters do not prevent *false positives*.

The most significant feature of Bloom filters is that there is a clear tradeoff between b and the probability of a false positive. Given h optimal hash functions, b bits for the Bloom filter and the number M of elements we can calculate the amount of false positives produced by the Bloom filter, as this will be explained in the next section.

4. BLOOMMAP ALGORITHM

In this section, we outline our *BloomMap Algorithm (BMA)*, which minimizes energy consumption and time overhead while guaranteeing location privacy. Instead of sending its RSS vector, the user forwards a Bloom filter, constructed from one Access Point (AP) in its vicinity, and its corresponding RSS value to the server. The server uses this Bloom filter to find a small number ($r << M$) of MATRIX rows that will allow the user to identify its location.

4.1 Outline of Operation

The basic BloomMap steps are the following:

1. The user selects a single AP-id from its vicinity to create a Bloom filter with user-defined redundancy k, according to the trade-off between the performance (i.e., size of the answer set) and privacy it chooses.
2. The user forwards the Bloom filter to the server.
3. The server finds the (k) AP-ids that could have been used to create this Bloom filter and retrieves the rows of the MATRIX, for which the RSS values are non-zero.
4. The server sends the resulting rows to the user in order to locally perform the localization process using the partial Radiomap of Step 3.

As discussed in the preliminaries, the user can define the redundancy of the Bloom filter and thus the number k of AP-ids it will map to, by selecting the number of bits b it will encode the AP-id in. We assume that the user knows a good approximation of the number M of AP-ids in the system and the h hash functions (e.g., these can be shipped from the server over initialization). In this way, we can ensure k-spatial anonymity, since the server will not be able to distinguish what APs the user can really listen to. The false positive ratio (*fpr*) is approximately given by the equation

$$fpr = (1 - e^{-hM/b})^h, \tag{1}$$

which means that given h, M and the user-defined k we can determine the number b of bits to use in the Bloom filter.

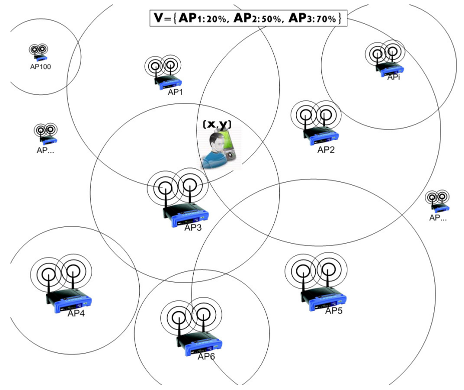


Figure 2: An example of a user's RSS vector V.

It is important to note that only one AP-id from the APs in the user's vicinity is used to create the Bloom filter. If more than one AP-id are used then the server will be able to identify the false positives (dummy AP-ids) given by the Bloom filter, since it will know the number of AP-ids used to create the Bloom filter and the random false positives will most likely not overlap. Thus, it would be impossible for a user to hear both APs.

The answer set can be further reduced with a slight tradeoff in the amount of cloaking achieved. Together with the Bloom filter, the user can also forward some RSS information to the server in order to match less rows from the radiomap. The amount of RSS detail corresponds to the amount of cloaking the users wish to have. Furthermore, the process on the server side can be speeded up by employing a variety of techniques, such as, an index on the AP-ids to the rows that have non-zero RSS values for the AP-id, or even low level bit operators to conduct the matching faster. These alternatives have not been exploited in this paper, but will be considered in the future when we deal with larger scenarios.

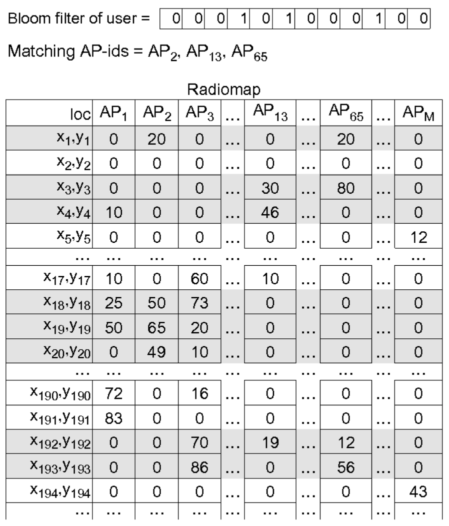
4.2 Running Example for BMA

As shown in Figure 2, consider a user at position (x, y) where signals of three different APs can be received. These APs have ids AP_1, AP_2 and AP_3, and their signal strength is 20%, 50% and 70%, respectively. Given a total number of $M = 100$ APs, a predefined set of $h = 3$ hash functions and a user's RSS vector $V = \{AP_1 : 20\%, AP_2 : 50\%, AP_3 : 70\%\}$, assume that a user wants to use $k = 3$ spatial anonymity.

The user first calculates the needed size b of the Bloom filter that will ensure $k-1$ false positives on the server-side. With $k = 3$ and $M = 100$, the false positive ratio of the Bloom filter is calculated to $fpr = k/M = 0.03$. Using the following false positive ratio equation,

$$b = \frac{-hM}{ln(1 - \sqrt[h]{fpr})}, \tag{2}$$

the user calculates $b \approx 12.5$ and sets $b = 12$ to ensure at least $k - 1$ false positives. Then the user chooses an AP_{id} from the RSS vector, e.g., AP_2, and feeds it to the predefined hash functions of the Bloom filter to generate a vector of size $b = 12$ with some bits turned to 1, e.g., $\{0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0\}$. It then sends the Bloom filter (or a lossless compression thereof) and the RSS value of AP_2 ($RSS = 50$) to the server.



Bloom filter of user = | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

Matching AP-ids = AP_2, AP_{13}, AP_{65}

Radiomap

loc	AP_1	AP_2	AP_3	...	AP_{13}	...	AP_{65}	...	AP_M
x_1,y_1	0	20	0	...	0	...	20	...	0
x_2,y_2	0	0	0	...	0	...	0	...	0
x_3,y_3	0	0	0	...	30	...	80	...	0
x_4,y_4	10	0	0	...	46	...	0	...	0
x_5,y_5	0	0	0	...	0	...	0	...	12
...	...	...	...	...	...	...	...	...	...
x_{17},y_{17}	10	0	60	...	10	...	0	...	0
x_{18},y_{18}	25	50	73	...	0	...	0	...	0
x_{19},y_{19}	50	65	20	...	0	...	0	...	0
x_{20},y_{20}	0	49	10	...	0	...	0	...	0
...	...	...	...	...	...	...	...	...	...
x_{190},y_{190}	72	0	16	...	0	...	0	...	0
x_{191},y_{191}	83	0	0	...	0	...	0	...	0
x_{192},y_{192}	0	0	70	...	19	...	12	...	0
x_{193},y_{193}	0	0	86	...	0	...	56	...	0
x_{194},y_{194}	0	0	0	...	0	...	0	...	43
...	...	...	...	...	...	...	...	...	...

Figure 3: Identifying MATRIX rows using the bloom filter at the server side.

On the server side, see Figure 3, the Bloom filter is used to identify the AP-ids that would produce the same Bloom filter and according to the chosen *fpr* it will match with at least $k = 3$ AP-ids on the server. One of those AP-ids, i.e., AP_2, is the actual AP-id that the user has in its RSS vector, see Figure 2. The other two, i.e., AP_{13} and AP_{65}, are the dummy elements that guarantee k-spatial-anonymity, letting the server know only with a probability of $1/k$ that the user is in the proximity of one of the matched APs. Notice that cloaking is also achieved, since the server does not just need to distinguish between $k = 3$ specific locations but $k = 3$ areas, where the user has equal possibility to be at any point inside those areas. Even if the server uses the RSS given by the user it will only be able to narrow down these areas, but not get 3 specific locations.

The server now retrieves for each matched AP the rows that contain a non-zero RSS value for the matched AP (denoted as shaded rows in Figure 3). The retrieved rows are finally send back to the user, who can now find its location by performing the mapping using one known algorithm e.g., KNN, WKNN, MAP, RBF, SNAP, etc., as in the DRA approach.

4.3 Discussion

The characteristics of the BloomMap algorithm are:

- **Energy Consumption (Good):** Only a very small subset of MATRIX rows are sent to the to user, ($r << N$).
- **Retrieval Time (Good):** Transmitting a small part of the radiomap ensures a fast reception of the results.
- **Privacy (Good):** Simple cloaking and k-spatial anonymity ensure that the server can only identify a wider area of the user's location with a probability less than 1/k.

The user can choose the amount of RSS details to send to the server. The more details the server receives the smaller the cloaking, but the fewer the resulting MATRIX rows will be. The user can choose to send just a range in which one or more of its RSS values lie, or the actual value of one or more of its RSS, and denote whether the RSS value or range corresponds to the AP-id inside the Bloom filter. Remember, that the amount of RSS details sent to the server does not affect the k-spatial anonymity guarantee. Regarding updates, our algorithm applies the same k-anonymity principle in the temporal dimension in order to provide additional cloaking, preventing the server to derive the user's location.

5. EXPERIMENTAL EVALUATION

In this section, we present a preliminary experimental evaluation of the BloomMap algorithm. We start-out with a description of our RSS prototype localization approach, i.e., Airplace. We then describe our experimental methodology and we finally proceed with the presentation of our results.

5.1 Our Airplace Localization System

The Airplace prototype architecture (see Fig. 1 (center)) consists of the *RSS Logger*, the *Find Me* application and the *Distribution Server*. These modules do not currently integrate BloomMap but presenting them allow us to put the BMA ideas in a real context.

Airplace RSS Logger Application: This application is developed around the Android RSS API for scanning and recording data samples in specific locations at predefined intervals. These samples contain the MAC addresses and RSS levels (in dBm) of all neighboring WiFi APs, as well as the coordinates of the location where the user initiated the recording.

Airplace Find Me Application: This is a client that runs on Android smartphones and connects to the server in order to download the Radiomap and algorithm-specific parameters in a single communication, thus enabling the user to self-locate independently thereafter. The interface of the *Find Me* application is shown in Fig. 4 (left), where the user can set the preferences (i.e., floorplan map, Radiomap download settings, etc.) and select any of the available positioning methods from the *Algorithms* configuration panel, which includes the deterministic K-Nearest Neighbor (KNN) and Weighted K-Nearest Neighbor (WKNN) algorithms [1], as well as the probabilistic Maximum A Posteriori (MAP) and Minimum Mean Square Error (MMSE) algorithms [19]. The Airplace platform additionally supports two state-of-the-art methods developed in-house, namely the Radial Basis Function (RBF) networks [12] and the Subtract on Negative Add on Positive (SNAP) [13] approaches.

Distribution Server: This server is mainly responsible for the construction and distribution of the RSS Radiomap by listening for connections from clients that either contribute the collected RSS data (wardriving) or request the Radiomap and algorithm parameters to start positioning. To create the radiomap file the server parses all available RSS log files, which may be contributed by several users, and calculates the mean RSS value per MAC address by averaging over all samples collected at each distinct location. Finally, all averaged values are merged and stored in a single Radiomap file so that each line corresponds to the mean RSS values at a specific location and each column to the respective MAC address.

5.2 Methodology

Datasets: We used the following two representative datasets of local sizes for our preliminary evaluation:

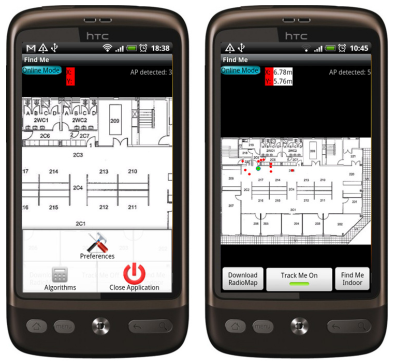


Figure 4: The *Airplace* positioning application.

i) UCY RadioMap: This radiomap is designed by data collected in a typical building at the Computer Science (CS) department of the University of Cyprus. We used three Android smartphone devices (HTC Hero, HTC Desire, Samsung Nexus S) to collect 30 RSS fingerprints (i.e., RSS values of APs at a reference location) at 1500 distinct locations for a total of 45,000 reference fingerprints. There are 120 WLAN APs installed in the four (4) floors of this building including the APs of neighboring buildings that can be partially detected in different sections of the CS building. On average, 10.6 APs are detected per location. We collected our data by walking over a path that consists of 2900 locations.
ii) KIOS RadioMap: We collected the RSS data inside a typical office environment at the KIOS Research Center, University of Cyprus. The total area is around $560m^2$ and the floor consists of several open cubicle-style and private offices, labs, a conference room and corridors. We have installed 9 APs that use the IEEE 802.11b/g standard to provide full coverage throughout the floor. In addition, there are several neighboring APs that can be partially detected in different sections of the floor. We used 3 devices running Android (HTC Desire, HTC Flyer, Samsung Nexus S) to collect RSS measurements from all available APs at 105 distinct reference locations. These locations are separated by 2-3 meters and form a grid that covers all public places. A total of 2100 training fingerprints, corresponding to 20 fingerprints per reference location, were collected at the rate of 1 sample/sec. For testing purposes, we have also collected 960 fingerprints at 96 locations (10 fingerprints per location) inside the experimentation area.

Algorithms and Metrics: We compare the BloomMap (BMA), DRA and CRA approaches, under a variety of settings using the datasets described earlier. Our cost metrics are: Time (T), Number of Messages (NoM) and Energy (E). All measurements are averaged over 10 consecutive runs.

Network and Energy Model: In our setting, a simple CRA REQ message is 4 bytes, a DATA message (i.e., a radiomap line) consists of each AP's MAC address that occupies 17 bytes and a 6-bytes field of it's RSS value, a DRA response that includes the client's actual location is of 10 Bytes. The communication between the smartphone and the server is performed under a 802.11b network link that has a TCP downlink of 1022kbps and a TCP uplink of 123kbps, a 237ms TCP handshake latency and application handshake

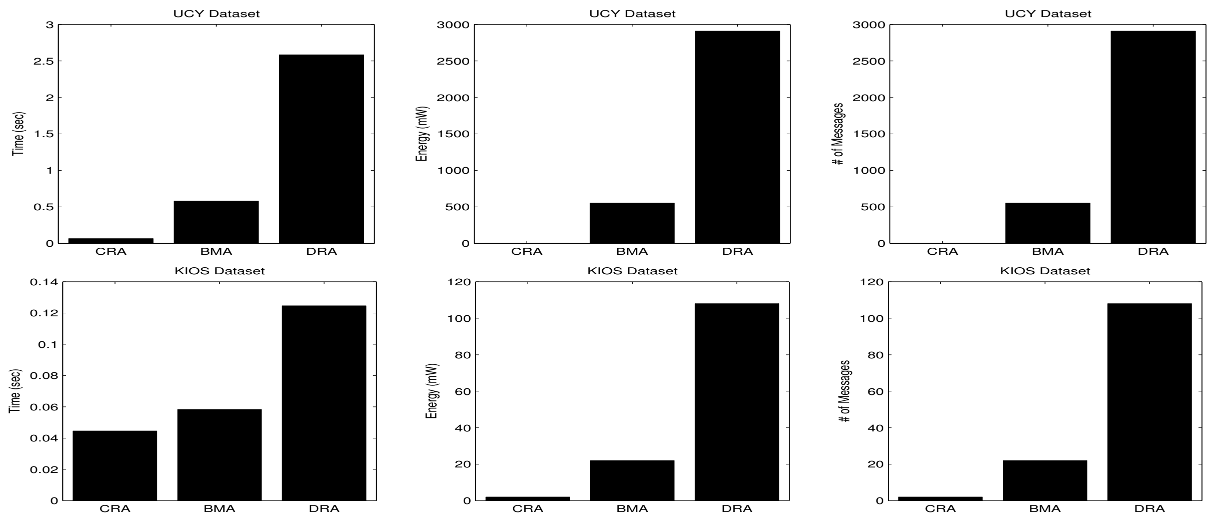


Figure 5: Preliminary evaluation of BloomMap algorithm on UCY and KIOS datasets.

latency of 493ms. Our energy profile has been derived by running test instances using PowerTutor [14]. In particular, CPU Idle (OS running) = 175mW, CPU Busy (Processing) = 369mW, WiFi Idle (Connected) = 38mW and WiFi Busy (Uplink 123Kbps, -58dBm) = 600mW.

5.3 Preliminary Results

Figure 5 shows the results of the three approaches in terms of Energy, Time and NoM for localization at a single reference location using the UCY and KIOS datasets described earlier. In the UCY dataset, the BloomMap algorithm improves the performance of the DRA approach (currently used in Airplace) by 80% in terms of time, 83% in terms of energy consumption and utilizes 80% less network resources. Similarly in the KIOS dataset, BloomMap algorithm provides 60% less time overhead and 60% less energy consumption and utilizes 80% less network resources. The CRA approach outperforms the BloomMap algorithm in both datasets. However, the CRA approach violates the user's privacy, where the BloomMap approach guarantees localization without revealing the user's real position.

6. FUTURE WORK

We are in the process of collecting larger-scale Radiomaps to test the scalability and robustness of our approach. Subsequently, we are going to exhaustively evaluate our algorithm in terms of other performance metrics (e.g., scalability) and compare it with other approaches.

Acknowledgments: This work was supported by the last author's startup grant, funded by the University of Cyprus.

7. REFERENCES

[1] P. Bahl and V. Padmanabhan. RADAR: an in-building RF-based user location and tracking system. In *INFOCOM*, pages 775–784, 2000.

[2] B. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of ACM*, 13(7), 1970.

[3] G. Chatzimilioudis, D. Zeinalipour-Yazti, W.-C. Lee, and M. Dikaiakos. Continuous all k-nearest neighbor querying in smartphone networks. In *MDM*, 2012.

[4] C.-Y. Chow and X. L. M.F. Mokbel. Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. In *Geoinformatica*, 2011.

[5] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K. Tan. Private queries in location based services: Anonymizers are not necessary. In *ACM SIGMOD*, 2008.

[6] Google. Google latitude, 2010.

[7] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *MobiSys*, 2003.

[8] A. Khoshgozaran and C. Shahabi. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In *SSTD*, 2007.

[9] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. In *IEEE ICP*, 2005.

[10] A. Konstantinidis, C. Aplitsiotis, and D. Zeinalipour-Yazti. Smartp2p: A multiobjective framework for finding social content in p2p smartphone networks. In *MDM*, 2012.

[11] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, and C. Panayiotou. The airplace indoor positioning platform for android smartphones. In *MDM*, 2012.

[12] C. Laoudias, P. Kemppi, and C. G. Panayiotou. Localization using radial basis function networks and signal strength fingerprints in WLAN. In *Globecom*, 2009.

[13] C. Laoudias, M. P. Michaelides, and C. G. Panayiotou. Fault tolerant fingerprint-based positioning. In *ICC*, 2011.

[14] PowerTutor.org. Powertutor: A power monitor for android-based mobile platforms.

[15] P. Samarati. Protecting respondents' identities in microdata release. *IEEE TKDE*, 13(6), 2001.

[16] L. Sweeney. k-anonymity: a model for protecting privacy. *Uncertain. Fuzziness Knowledge-based Syst.*, 10(5), 2002.

[17] M. Yiu, G. Ghinita, C. Jensen, and P. Kalnis. Outsourcing search services on private spatial data. In *ICDE*, 2009.

[18] M. Yiu, C. Jensen, X. Huang, and H. Lu. Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In *ICDE*, 2008.

[19] M. Youssef and A. Agrawala. The Horus WLAN location determination system. In *MobiSys*, pages 205–218, 2005.

[20] D. Zeinalipour-Yazti, C. Laoudias, C. Costa, M. Vlachos, M. I. Andreou, and D. Gunopulos. Crowdsourced trajectory similarity with smartphones. *IEEE TKDE*, 2012.