RadioMap Prefetching for Indoor Navigation in Intermittently Connected Wi-Fi Networks

Andreas Konstantinidis*, George Nikolaides*, Georgios Chatzimilioudis*, Giannis Evagorou‡, Demetrios Zeinalipour-Yazti* and Panos K. Chrysanthis§

*Department of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus
‡Department of Computing, Imperial College London, SW7 2AZ London, UK
§Department of Computer Science, University of Pittsburgh, PA 15260, USA
{akonstan, gnik, gchatzim}@cs.ucy.ac.cy; giannis.evagorou.14@ucl.ac.uk; dzeina@cs.ucy.ac.cy; panos@cs.pitt.edu

***Abstract*—Wi-Fi (or WLAN) based indoor navigation applications for mobiles rely on cloud-based *services (s)* that take care of a *user's (u)* localization task using structures called *RadioMaps (RMs)*. It is imperative for *u* to have a stable Wi-Fi connection in order to either continuously receive location updates from *s* or to download *RMs* a priori for offline navigation. Wi-Fi networks however, suffer from *intermittent connectivity* due to poor network planning that results in sparse deployment of access points and effectively areas where Wi-Fi coverage cannot be guaranteed. This inherently affects the localization accuracy and therefore the navigation experience of users. In this paper, we propose an innovative framework for accurate and fast indoor localization over an intermittently connected Wi-Fi network, coined *Prefetching Localization (PreLoc)*. In *Preloc*, we prioritize the download of *RM* records based on knowledge acquired from historic traces of other users inside the same building. Instead of downloading the complete *RM* from s to u, we propose a *Probabilistic Group Selection (PGS)* strategy, which identifies RM records that have a higher probability of being necessary to a user moving inside a target area. We have evaluated our framework using a real prototype developed in Android, as well as realistic Wi-Fi traces we collected at the University of Cyprus. Our experimental study reveals that *PreLoc* using *PGS* and conventional fingerprint-based indoor positioning algorithms can yield accuracy that is as good as using the same algorithms with a complete *RM*, even under scenarios of weak Wi-Fi coverage.**

Keywords—Indoor Localization, Prefetching, Radiomaps

I. Introduction

People spend 80-90% of their time in indoor environments[1], including shopping malls, libraries, airports or university campuses. The omni-present availability of sensor-rich mobiles has boosted the interest for a variety of indoor location-based services, such as, in-building guidance and navigation, inventory management, marketing and elderly support through Ambient and Assisted Living [1][2].

To enable such indoor applications in an energy-efficient manner and without expensive additional hardware, modern smartphones rely on cloud-based *Indoor Positioning Services (IPS)*, which either provide the accurate location (position) of a user upon request or provide structures for offline localization and navigation. There are numerous IPS, including *Skyhook-Wireless.com, Google.com, Navizon.com, Infsoft.com, Indoo.rs, IndoorAtlas.com* and our in-house *Anyplace*[2] service [3]. These IPS services rely on cloud-based geolocation databases that maintain information about Wi-Fi Access Points and Cellular Towers (i.e., the signal intensity of these transmitters at known locations in space). These IPS geolocation database entries act as reference points for requested localization tasks, as explained thoroughly in Section II. In summary, a smartphone can determine its location at a coarse granularity (i.e., km or hundreds of meters) up to a fine granularity (i.e., 1-2 meters) [4], by comparing against the reference points stored in the geolocation database.

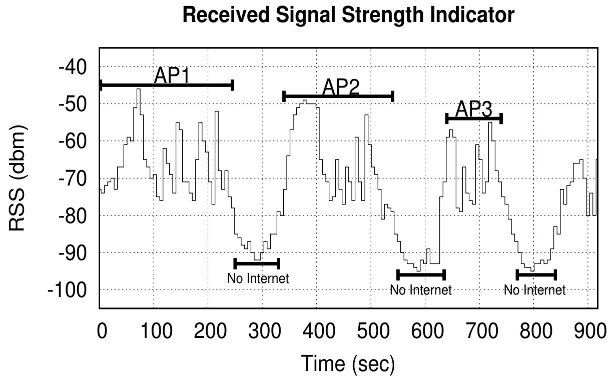


Fig. 1. A user navigates inside a mall. Due to the Wi-Fi network arrangement, the user gets disconnected from the network at several points in space. This effectively hinders the accurate localization of the user that relies on a cloud service for its location updates.

One fundamental drawback with IPS is that mobile users need to communicate with the service over a Wi-Fi network while using it. Unfortunately, Wi-Fi networks suffer from *intermittent connectivity* [5], i.e., Wi-Fi coverage is irregularly available inside a building usually due to poor network planning. This inherently affects the localization accuracy and effectively the navigation experience of users. For example, Figure 1 illustrates the Wi-Fi connectivity of a smartphone device while navigating inside a real mall. In particular, a smartphone user measures the *Received Signal Strength (RSS)* from the *Access Point (AP)* to which it is connected while being on the move. Particularly, the user experiences periods where the smartphone device is disconnected (these often occur when the RSS is below a certain threshold, which is around -85 dBm) before it reconnects to the next AP.

[1]US Environmental Protection Agency, http://epa.gov/iaq/
[2]Available at: http://anyplace.cs.ucy.ac.cy/

One could claim that alternative connection modalities (e.g., 2G, 3G or 4G), could have been used to offset the drawbacks related to limited Wi-Fi coverage (e.g., to fallback to mobile internet when necessary). However, mobile internet alternatives have their own limitations, even if we assumed that everybody had one, including: (i) *limited coverage*, due to the blockage or attenuation of the cellular signals inside buildings these are not available in deep indoor spaces; (ii) *limited quota*, due to the fact that mobile internet might have monthly quotas that make users unwilling to consume it; (iii) *unavailability due to roaming*, given that many users turn their mobile internet roaming off, when traveling outside their telecom operator, in order to avoid excessive monetary charges; and (iv) *slow fallback* from Mobile Internet to Wi-Fi and vice-versa, up to several tens of seconds, due to a slow transition process that mainly relates to old Wireless and Mobile standards (e.g., the lack of ubiquitous 802.11k, 802.11r and 4G infrastructure).

The assumption of Internet connectivity for localization in outdoor environments using satellite-based *Global Navigation Satellite System (GNSS)* does not exist, as these perform the localization directly on the terminal with no mandatory location information downloaded from any type of service. On the other hand though, GNSS have an expensive energy tag that is negatively affected by the environment (e.g., cloudy days, forests, downtown areas) and requires an unobstructed line-of-sight to GNSS satellites making it insufficient for indoor localization scenarios. Although this work mainly concerns fine-grain localization scenarios in indoor spaces, our discussion is equally applicable to coarse-grain localization scenarios in urban outdoor spaces using Wi-Fi, but we will not consider these scenarios in this paper.

In this paper, we consider that the communication between smartphone users and IPS suffers from *intermittent connectivity* and, as such, develop hybrid techniques that exploit the IPS utility and maintain the localization accuracy continuously. Particularly, we tackle the following technical challenge:

*"Assuming a mobile indoor user (*u*) and a network that suffers from intermittent connectivity, thus a finite amount of time until* u *gets disconnected, how can the download of* RMs *be prioritized in order to enable* u *to localize continuously and accurately with a low computational effort?"*

We devise the *Prefetching Localization (PreLoc)* framework, which guarantees that u will continuously localize, and therefore navigate in an indoor environment, by selectively prefetching a subset of localization entries from s without deteriorating the system's performance. Our framework is suitable for intelligent navigation systems [6], location-aware social networking and guidance applications for smartphones [7] and others. An indicative scenario supported by our framework might be: *"Navigate from the central bus station to Gate 29 of Terminal 1 inside Heathrow airport"*, or *"Find the Data Management Systems Laboratory at the University of Cyprus"*.

As an example, consider the illustration of Figure 2 (left). An arbitrary user u moves between discrete time steps among points towards building A and localizes itself at each point using the *PreLoc* smartphone application. While u requests a localization from s to navigate through building A, s computes its location and selects a number of location reference entries, which u prefetches. When u experiences disconnection (shadowed area in figure) but still needs to localize, it uses the prefetched entries that are cached locally to localize itself and navigate to its destination, as shown in Figure 2 (right).

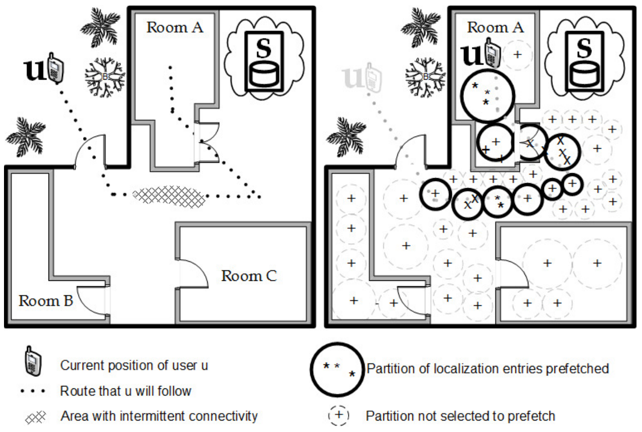

Room A
S
u
Room B
Room C
Current position of user u
Route that u will follow
Area with intermittent connectivity
Partition of localization entries prefetched
Partition not selected to prefetch

Fig. 2. (Left) Indoor localization of user u using the cloud-based IPS s over a network that suffers from intermittent connectivity (shaded area). (Right) The RM is clustered and partitioned using our *PreLoc* framework, which subsequently selects the most appropriate clusters using our proposed *PGS* heuristic. Those clusters are finally prefetched by u from s.

Particularly, *PreLoc* operates in three phases: (i) the *pre-processing phase*, where s partitions the available reference locations of a particular area in an offline manner; (ii) the *selection and prefetching phase*, where s carefully selects a small set of partitions and u prefetches the selected partitions on its device; and (iii) the *localization phase*, where u localizes itself using a well known localization technique, e.g., NN, KNN [8] or Weighted KNN [9], and the available prefetched location data. For phase (ii), we propose an efficient technique, coined *Probabilistic Group Selection (PGS)*, to select the location reference entries that are most likely to be needed by the user. This is done based on historical users' movement data. In cases of intermittent connectivity, u localizes itself with any prefetched partitions that adequately cover a relatively long user route without requiring further location information.

Our contributions in this work are summarized as follows:

- We propose an innovative framework, coined *PreLoc*, for accurately localizing and navigating in an indoor space under intermittent Wi-Fi connectivity issues.
- We propose a heuristic for prefetching *RM* entries, coined *Probability Group Selection* (*PGS*), which uses historical mobility data collected, clustered and organized in an offline phase.
- We confirm the efficiency of our propositions using a real prototype system implemented in Android and real collected data on a 8,900 m^2 university campus. Our study reveals that *PreLoc* can yield excellent accuracy even under scenarios of low Wi-Fi coverage.

The remainder of the paper is organized as follows: Section II provides the related work while Section III our system model and problem formulation. Section IV presents the *PreLoc* framework and describes the proposed techniques that compose it. Section V presents our experimental methodology and results, while Section VI concludes the paper.

II. Background and Related Work

In this section, we provide background and related work on indoor localization as well as intermittent connectivity in Wi-Fi networks, upon which our propositions are founded.

A. Background on Wi-Fi Indoor Localization

The localization literature is very broad and diverse as it exploits several technologies. GNSS (e.g., GPS) is obviously ubiquitously available but has an expensive energy tag and is also negatively affected from the environment. Besides GNSS, the localization community [1], [10], [11] proposed numerous proprietary solutions including: *Infrared, Bluetooth, visual or acoustic analysis, RFID, Inertial Measurement Units, Ultra-Wide-Band, Sensor Networks, Wireless LANs, etc.*; including their combinations into hybrid systems. Most of these technologies deliver a high level of positioning accuracy, however they require the deployment and calibration of expensive equipment, such as custom transmitters, antennas or beacons, which are dedicated to positioning. This is time consuming and implies high installation costs, while the approaches we discuss operate off-the-shelf on conventional smartphones.

There are also approaches that use radio signals from mobile Cell Towers, Wi-Fi APs, or their combination, to offer coarse accuracy that is often 10-30 meters. The signals are stored in databases constructed offline by contributors (e.g., an Android phone by default forwards Wi-Fi AP and Cell Tower data to Google). Subsequently, users can obtain their current location using a query / response to the cloud-based localization service. For this category, the localization is strongly influenced by intermittent connectivity due to the fact that u must continuously exchange information with s for every single localization.

Radiomap-based approaches are similar to the latter approaches [12], but at a much higher density [13]. These might additionally introduce error due to device diversity reasons [14] and the human factor [15]. For example, our Anyplace [3] and open-source Airplace [16] systems, achieve the second highest known accuracy [4] with an average error of 1.96 meters that works as follows: in an offline phase, a logging application records the so called *Wi-Fi fingerprints*, which comprise of *Received Signal Strength RSS* values of Wi-Fi AP at certain locations (x,y) pin-pointed on a building floor map (e.g., every few meters) as well as orientation data. Subsequently, in a second offline phase, the Wi-Fi fingerprints are joint into a NxM matrix, coined the *Wi-Fi RM*, where N is the number of unique (x,y) fingerprints and M the total number of APs. Finally, a user can compare its currently observed RSS fingerprint against the RM in order to find the best match, using known algorithms such as NN, KNN [8] or Weighted KNN [9].

Particularly, the K-Nearest-Neighbor (KNN) approach calculates the Euclidean distance d_i between the user u's currently observed fingerprint V_u against all fingerprints V_i in the RadioMap, i.e., $d_i = ||V_i - V_u||, \forall V_i \in RM$. Then the K nearest fingerprints around the user's device are selected and the user is positioned using convex combination of those K locations. However, by considering that all K nearest neighbor fingerprints are of equal importance (i.e., assigned an equal weight equal to $w_i = 1/K$) may decrease the localization accuracy, since fingerprints that are far away may also be included in the calculation. Therefore, a more effective way of weighting the K nearest fingerprints is required. In the Weighted-KNN (WKNN) approach, the K nearest neighbors, calculated as in KNN, are assigned a weight equal to:

$$w_i \propto \frac{1}{||V_i - V_u||}$$

Finally, the user's location is calculated again using a convex combination of those K locations, where in this case the farther locations affect less the calculation than the closer locations.

Discussion: For the final RSS fingerprint comparison step, we differentiate between the following two cases: a) Wi-Fi RadioMap *Server-Side Approach (SSA)*, where the localization takes place on the IPS and which is considered as the base approach in this work; and b) Wi-Fi RadioMap *Client-Side Approach (CSA)*, where the RadioMap is downloaded to the smartphone prior the localization. In *SSA*, localization can be achieved with little network messaging and minimal energy consumption, as the bulk of operation takes place on the IPS that has an unlimited energy and processing budget. Unfortunately, since the localization in *SSA* is carried out by the IPS, this approach fundamentally suffers from intermittent connectivity issues. *CSA* on the other hand, does not suffer from any network related issues, but unfortunately requires the download of the RadioMap. As RadioMaps can potentially be very large (e.g., WiGLE.net had 2.8 billion unique records by April, 2015), the *CSA* leads to high overhead time and the waste of precious and limited smartphone battery and bandwidth. This applies even if we assumed a finer granularity (e.g., a 8,900 m^2 building requires ≈3MB while a whole campus around 50-100MB). Our experimental evaluation in Section V validates this argument.

B. Mobile (Intermittent) Connectivity and Prefetching

In Wi-Fi networks, intermittent connectivity refers to the frequent disconnection of a mobile node in random time intervals. This often occurs due to the following two reasons [5]: (i) there is a gap between the coverage of two APs and thus the connectivity experienced by mobile users passing by will likely to be intermittent; and (ii) because of physical obstacles as well as high mobility patterns of the mobile users. In either case, intermittent connectivity may break data connections, if the connectivity disruption between a mobile node and an AP is long enough and the available transfer rate provided is below a certain threshold. Simple solutions, such as auto-correction or manual reconnection are not practical due to the excess overhead that this process requires.

Several techniques have been proposed to tackle the intermittent connectivity problem in mobile networks such as mobility management [17], cooperative downloading schemes [18], AP deployment algorithms [19], prefetching [20], routing [21], [22] or combinations of those techniques [5]. Prefetching systems in mobile networks aim at hiding the frequent disconnections and/or the latency of data transfers over poor and intermittently connected environments. In particular, a prefetching system predicts what data an application will request in the future and speculatively retrieves and caches that data in anticipation of those future needs [20].

Discussion: To the best of our knowledge there is no prior work that solves the problem of indoor localization in an environment with intermittent connectivity. Furthermore, there is no prior work in the literature that adopts a fundamental prefetching technique for improving indoor localization accuracy and optimizing resource consumption on smartphones. In this paper, we use prefetching for improving both localization accuracy and resource consumption on smartphones, at the same time, allowing the user to continuously localize without any disruptions by prefetching and caching reference locations.

III. System Overview

This section presents the assumptions, system model and problem formulation of our work. The main symbols used throughout the paper are summarized in Table I.

A. Indoor Localization using Wi-Fi fingerprints

We assume a planar indoor area I containing a finite set of locations that are partially covered by a set of Wi-Fi Access Points $AP = \{ap_1, ap_2, \cdots, ap_M\}$ (see Figure 3). Each ap_i has a unique ID (i.e., MAC address) that is publicly broadcast and passively received by anyone moving in the coverage of ap_i.

The signal intensity at which the ID of ap_i is received at location l, is termed the *Received Signal Strength (RSS)* of ap_i at l, having a value in the normalized range $[0..100]$. The set of RSS values measured and the ap-IDs read at a location l is termed *fingerprint* V_l of location l. A user u at l attempts to automatically connect to exactly one $ap_i \in AP$ in its vicinity, using a typical but slow transition process or designated standards (e.g., 802.11k and 802.11r).

We further assume an indoor positioning server s that has constructed beforehand a *RadioMap* (RM). RM maps known locations $l \in I$ (rows) to the *fingerprint* V_l measured at each location, where a value of -1 indicates that an ap_i is out of reach. Any subset of RM rows will be denoted as *partial RadioMap* (RM_i). Server s uses a localization function *loc()*, which can compute an estimation $\lambda_{l'}$ of location l' given RM or RM_i and the fingerprint $V_{l'}$ at l'. Note that l' does not have to be a location recorded within RM.

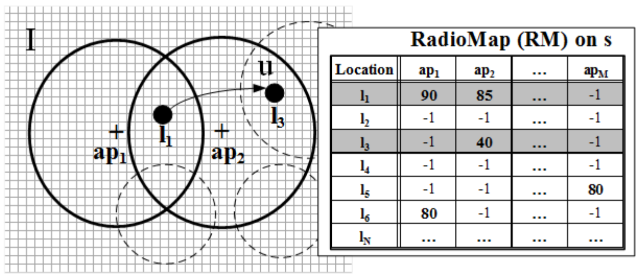
B. Connectivity Probability in Indoor Environments

In line with the above assumption that the arrangement of Wi-Fi APs in I results in partial coverage and weak RSS at some locations, we define an RSS threshold θ, below which the data transmission rate of a mobile user is practically zero. Specifically, client u with a fingerprint V_r at timestep of request r is practically offline if the maximum signal strength max_{V_r} it receives from the most powerful covering ap_i is below threshold θ, i.e., $max_{V_r} < \theta$.

Based on this threshold we define the *Connectivity Probability* P_I of space I with which a user u succeeds to receive an answer to its localization request r through s. For ease of exposition we will assume that a localization request r is answered by s instantly, therefore, P_I denotes the probability in I that u measures an RSS above θ.

TABLE I. Notation used throughout this work

Notation	Description
I, l	Indoor space, Location inside I
ap_i, AP	Access Point i, Set of all ap_i
s, u, U	Positioning Service, User, Set of all u
V_x	fingerprint at location or time x (RSSs and ap-IDs)
RM, RM_i	RadioMap mapping l to V_l, Partial RadioMap
r, R	Timepoint of localization request, Set of all r
loc()	Localization function exploiting fingerprints and RM
l_r^u	Actual location of u for request r
λ_r^u	Estimation of l_r^u computed by *loc()*
θ	RSS threshold below which practically not connected
P_I	Connectivity Probability of space I
A_r	Point Accuracy achieved at timepoint r
α	A_r achieved by *loc()* when RM available
T_r	CPU Time cost of u for request r



RadioMap (RM) on s

Location	ap_1	ap_2	...	ap_M
l_1	90	85	...	-1
l_2	-1	-1	...	-1
l_3	-1	40	...	-1
l_4	-1	-1	...	-1
l_5	-1	-1	...	80
l_6	80	-1	...	-1
l_N	...	...	...	...

Fig. 3. System Model: i) user u moving in area I partially covered by access point set AP, requests localization from s; and ii) a RadioMap RM on s.

Connectivity Probability (P_I) *is the probability that* u *has a signal strength above* θ *whenever it sends a localization request* $r \in R$, *and is given by*

$$P_I = \frac{\sum_R success(r)}{|R|} \tag{1}$$

$$\text{where } success(r) = \begin{cases} 0 & if \max_{V_r} \leq \theta, \\ 1 & if \max_{V_r} > \theta. \end{cases}$$

C. Research Goal and Metrics

Research Goal. *Enable a mobile user to consecutively localize itself accurately and efficiently in an indoor environment, where connectivity is intermittent, using an Indoor Positioning Server holding a RadioMap.*

The efficiency of the solutions proposed is measured by the *Point Accuracy* [4] achieved by the localization and the *CPU Time* needed on the client device.

Point Accuracy (A_r) *is the Euclidean error distance between the location estimation* λ_r^u *and the actual location* l_r^u *of user* u *at the timepoint of localization request* r, *and is given by*

$$A_r = |\lambda_r^u - l_r^u| \tag{2}$$

We assume that a localization function *loc()* achieves constant Point Accuracy α for each localization request r if it has access to the whole RM.

CPU Time (T_r) *is the processing time used on the device of u for running the localization function loc() given a request r and a subset RM_i.*

Our research goal can be expressed by the minimization of the following two objective functions:

$$\begin{aligned}\min F_A &= \frac{1}{|R|}\sum_R (A_r - \alpha)\\ \min F_T &= \frac{1}{|R|}\sum_R T_r\end{aligned} \tag{3}$$

D. Baseline Approaches

Existing techniques for indoor localization using Wi-Fi fingerprinting can be categorized as follows:

i) Server-side approach (SSA): User u measures the Wi-Fi fingerprint V_r at the timepoint of request r and sends it to server s. The location estimation λ_r^u of u is computed on s executing *loc*(V_r, RM) and transmitted back to u. In this scenario, u sends V_u and receives λ_r^u, without performing any further computation. Therefore, the value of objective function F_T is *minimum* (i.e., 0).

Drawback: This method suffers from intermittent connectivity, since the Point Accuracy A_r depends on the constant communication with s. In the case of successful communication, an optimal Point Accuracy $A_r = \alpha$ can be achieved by *loc()* using RM on s. Otherwise, the best estimation that can be made about the location of u at timepoint r is the last localization computed by s. In this case, the Point Accuracy is calculated by $A_r = |\lambda_r'^u - l_r^u|$, which is the difference between the actual location l_r^u of u at the current request r and the location estimation $\lambda_r'^u$ at the last request r' where u managed to communicate with s. Therefore, in the *SSA* approach the Point Accuracy grows worse when u moves further away from the last location of connectivity. This means that the smaller the Connectivity Probability P_I of an indoor space I is, the worse the objective function F_A for Point Accuracy *grows*.

ii) Client-side approach (CSA): User u downloads the whole RM from s before reaching an area not covered by Wi-Fi (at a building or floor granularity). Assuming that the download process has completed, u can now localize itself using the localization function *loc()* with its fingerprint V_r and RM as its input. This technique minimizes the objective function F_A since it achieves optimal Point Accuracy $A = \alpha$ and it is not affected by intermittent connectivity.

Drawback: The objective function F_T is undesirably *maximized* since u performs the localization using the complete RM locally. This results in the execution of a large number of unnecessary distance calculations between the current fingerprint and the RadioMap, i.e., $d_i = ||V_i - V_u||, \forall V_i \in RM$. For ease of exposition, we do not take into consideration the CPU Time needed to download RM given that this could have occurred in an offline phase.

IV. Prefetching Localization (*PreLoc*) Framework

In this section, we describe our *PreLoc* framework, present its underlying algorithms and techniques as well as their expected performance.

Algorithm 1 . PreLoc Framework Outline

```
Require: Selection Technique $selectEntries()$, Selection size $K$, RadioMap $RM$
Ensure: High accuracy localization as client moves
 1: client: $V_r$ = measure RSS fingerprint at current location
 2: if connected then
 3:     client: send $V_r$ to $s$
 4:     server: $\lambda_r^u = loc(V_r, RM)$
 5:             $RM_r^u = selectEntries(\lambda_r^u, RM, K)$
 6:             send $RM_r^u$ and $\lambda_r^u$ to $u$
 7:     client: update $RM^u$
 8: else
 9:     client: $\lambda_r^u = loc(V_r, RM^u)$
10: end if
```

A. Outline of Operation

The intuition behind the *PreLoc* framework is to prefetch a group of RadioMap entries RM_i that can aid localization at the client side u, in case u looses connection to the server s. Whenever u has connection to s, it requests localization and an RM_i. On the other hand, whenever u looses connection to s, it uses any locally stored RM_i to compute its own location.

Specifically, u transmits its measured fingerprint V_r to s. The server computes λ_r^u using localization function *loc()* that processes the V_r and operates on RM. It further selects a subset of entries RM_r^u to be sent to u for successfully localizing in a disconnected operation state. In the case where u can not communicate with s then u locally runs function *loc()* by processing its measured fingerprint V_r and any existing RadioMap entries RM^u prefetched from s earlier. Therefore it is obvious that the Point Accuracy A of localization in the disconnected case depends heavily on three things: (i) the entries RM_r^u prefetched at r, (ii) the number $K = |RM_r^u|$ of entries prefetched at request r, and (iii) how far u moved from the last connected location.

Next, we present techniques for selecting the entries to be prefetched and how these heuristics affect the Point Accuracy in the disconnected case, i.e., with intermittent connectivity.

B. Selection Technique PGS

The entries chosen to be sent to the client u greatly affect the Point Accuracy A of a disconnected localization state. Naive solutions include, selecting random entries or selecting the closest entries to the estimated location λ^u of client u.

We propose a heuristic for selecting K groups of entries from RM to be prefetched that are based on historical data of past user movements and therefore of higher probability that u will visit based on its current whereabouts. We coin this heuristic *Probability Grouping Selection (PGS).*

To compute the probability of visiting a particular location, we use historical mobility data collected by s to construct a *Dependency Graph* (DG) between entries. This approach is inspired by Web prefetching methodologies [23], [24]. DG represents the motion habits of clients in the given area. The graph contains a node for each entry of the area while the edges represent the transitions from one entry to another. Our graph is trained using several lookahead window sizes W [25].

The Dependency Graph (DG) is constructed for the provision of hints for the selection process. The DG algorithm constructs a DG that includes a node for each RM entry that has been accessed by other clients in the past. Assume two entries C_1, C_2, and an edge $C_1 \rightarrow C_2$ on the graph if and only if a client has been localized within entry C_2 after entry C_1. It should be noted that the path between entry C_1 and C_2 may contain some intermediate entries. The edge weight is the ratio of the number of accesses of C_2 after C_1.

The construction algorithm consists of three phases. In the first phase a pass over all trajectories is being made that populates the node set and calculates the frequency of each node. In the second pass, edges of various depth are added to the edge set. As a result, the frequency of each transition for different lookahead windows W is increased. Finally, the weights (transition probabilities from one entry to another) of all arcs are calculated.

C. Selection size K

The larger the value for K, the larger the area where a client u will localize with good Point Accuracy A while being disconnected. As K increases we achieve better accuracy A, since there is a bigger likelihood that the client locally finds an entry. On the other hand, as K increases, more CPU Time T is needed for processing a larger partial RadioMap RM_i. Also, prefetching will be more expensive, since more entries need to be downloaded every time u connects to s. Therefore, the optimal value for K depends on the Connectivity Probability P_I of a building. Therefore, there is a tradeoff between CPU Time T and Point Accuracy A.

For the ideal selection algorithm, where only the entries that will actually be needed during a disconnection period are selected, and assuming the ideal probabilistic model where u is connected in every $1/P$ requests, then the ideal number of entries to download is $K = 1/P$.

D. Localization Algorithm

Given an RSS fingerprint V_r and a partial RadioMap RM_i, we can localize using existing localization algorithms [16]. The output is a location estimation λ_r^u from where the given RSS fingerprint was measured. In our framework, we implemented the *Weighted K Nearest Neighbor* (WKNN) localization algorithm [9], where c neighboring entries of the fingerprint are weighed and used to compute the location.

When value K used for prefetching is smaller than the value of WkNN parameter c, the accuracy A_r when disconnected will always be worse than when connected, even if the entries prefetched are the ideal ones. Furthermore, using the *PGS* selection strategy that is trying to match user movement, the resulting RM_r^u will contain K entries that are more likely to be spread out linear rather than around the location of the user. Therefore, even when K is larger than c the accuracy A_r when disconnected will be worse than when connected.

E. Partitioning the RadioMap

For any localization algorithm to work efficiently, more than one *RM* entry is needed around the location to be computed. Furthermore, *RMs* in the real world tend to be very large with hundreds or even thousands of geographically dense entries. For these reasons we partition the *RM* entries into groups and run our selection and localization algorithms on these groups rather than individual entries. This results in faster computation and more precise localization. Furthermore, dealing with *RM* partitions instead of single *RM* entries allows more efficient communication (paging) and prefetching (less distances computed).

Any partitioning or clustering algorithm can be used for this preprocessing step [26]. There are two types of partitions that can be achieved: equi-width and equi-depth [27]. In the former, each partition will correspond to an area of approximately the same size and will contain all *RM* entries within this area. The advantage of this partitioning type is that the number of partitions prefetched directly determines the area where the user can navigate without connection and therefore determines the Point Accuracy A achieved. The drawback is that the entry population of each partition might vary greatly, resulting in great variation in the CPU cost of the client device. In the latter partitioning type, each partition will have approximately the same entry population. In this case we have the opposite advantages and drawbacks.

In this work, we use the BFR equi-width partitioning algorithm [28] for partitioning *RM*. BFR is a variant of the k-means algorithm and is specifically designed to handle very large disk-resident data sets. Points are read per block, where the block size is the available space in main-memory. The points from the previous block are summarized by simple statistics. Using the first block that arrives in memory we select L centroids in space. We implemented BFR with Mahallanobis distance [29], because it is (i) less susceptible to outliers than closest or farthest point distance, (ii) faster to compute than the all-point average distance, and (iii) unit-less, scale-invariant, and takes into account the correlations of the data set.

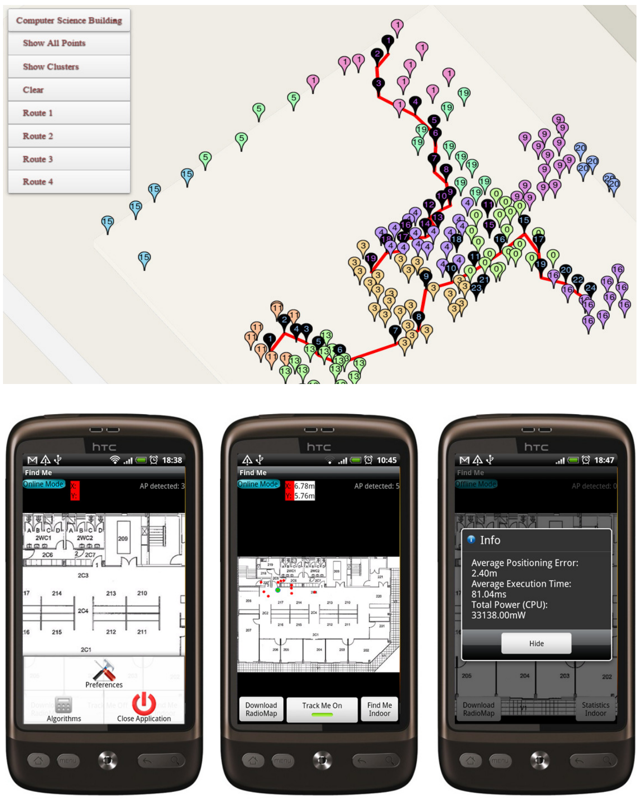
Greater values for parameter L result in smaller partitions, which need less communication cost to be downloaded by the client and less CPU Time to be processed during localization. The disadvantage of having smaller partitions is that the K clusters downloaded cover a smaller area, therefore, the Point Accuracy drops whenever the client looses connection and moves away from that area.

V. Experimental Evaluation

This section presents an extensive experimental evaluation of the *PreLoc* framework and the proposed RM entry selection techniques. We start-out with our experimental methodology and setup followed by our experimental series.

A. Methodology

CSUCY Dataset: this dataset was collected at the Computer Science (CS) department of the University of Cyprus (UCY). The building is around 8,900 m^2 and has four floors (i.e., 2,224 m^2 per floor). In total, it consists of 45,000 reference fingerprints taken from ≈120 Wi-Fi APs (wired or hotspots) installed in the *CS* and neighboring buildings. On average, 10.6 APs are detected per location. We collected our data by walking over a path that consists of 2,900 locations. The CSUCY normalized Radiomap has a size of ≈2.6 MBs (the initial fingerprint database was much larger).



Average Positioning Error:
2.40m
Average Execution Time:
81.04ms
Total Power (CPU):
33138.00mW

Fig. 4. ***PreLoc* web-visualizer (top):** used for testing and verification of the partitions and routes generated for our simulations. Clusters are identified using different colors and number, where routes are indicated as sequence of numbers connected with solid lines. **Airplace [16] (bottom):** our in-house indoor localization and navigation platform extended to support the *PreLoc* framework functionalities.

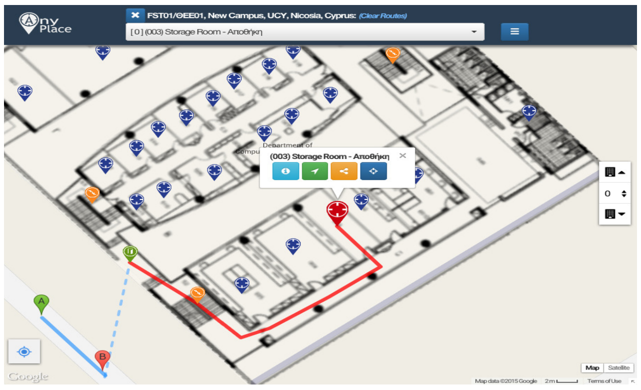

AnyPlace
FST01/ΘΕΕ01, New Campus, UCY, Nicosia, Cyprus: (Clear Routes)
[0] (003) Storage Room - Αποθήκη
(003) Storage Room - Αποθήκη
Map
Satellite
Google

Fig. 5. The AnyPlace Indoor Information Service (screenshot relates only to the Web Viewer) facilitates the task of crowdsourcing models, context and radiomaps for indoor spaces but also supports intelligent search and navigation in buildings uploaded publicly or privately.

User Routes and Partitioning: In order to evaluate the scalability of our propositions we have generated realistic user routes of various scales where a user follows and localizes at pre-defined locations. The distinct locations are of fixed distance between each other (e.g., around 5 meters) and the size of routes varies from 15-30 localizations steps (i.e., a user moving at a single floor and travels around 50-150 meters). The RadioMap is also partitioned into L partitions/groups using the BFR approach as described in Section IV. Both the routes and the partitions can be viewed and verified using our web visualizer that is described below.

Metrics and Algorithms: We are interested in localization accuracy and resource consumption on the client side, therefore, our cost metrics are: (i) the average *CPU Time* (T) on the client device; and (ii) the average *Point Accuracy* (A) achieved in an environment with intermittent connectivity P. These were measured over all localizations in all routes for varying values of P. We compare two existing prefetching approaches against our *PreLoc* framework. The existing approaches fall under the category of the *Client-Side Approach (CSA)* as described in Section II, where *CSA (b)* denotes the RadioMap of the whole building, while *CSA (f)* denotes the RadioMap of the current floor. The *PreLoc* framework is evaluated using three different RM entry selection techniques: (i) the proposed *Probability Group Selection* (*PGS*) technique as described in Section IV; (ii) the *Iterative Deepening Selection* (*IGS*) technique that selects partitions to be prefetched in an iterative manner based on the distance to current location; and (iii) the *Random Selection* (*RS*) technique that selects partitions to be prefetched randomly. To put the results in perspective when comparing to localization accuracy, we also used the *Server-Side Approach (SSA)* that was described in Section II, which does not prefetch any localization data and thus is prone to intermittent connectivity problems.

Web Visualizer: We have also developed a web application for visualization, validation and testing purposes of the partitions and routes considered in our experiments that follow. Figure 4 illustrates a snapshot of our in-house *PreLoc* web-visualizer that shows the partitioning of all available reference locations (fingerprints indicated with different letters) as well as the four routes (indicated as sequence of numbers connected by a solid line) used for the evaluation of our propositions.

Airplace/Anyplace: Finally, the *PreLoc* framework was built around our open-source indoor localization engine coined *Airplace* [16] (see Figure 4 bottom). It consists of two major components: the *Server* and the *Client* application for Android smartphones. The *Anyplace Client* has two modes and operates either as a *Logger* or as a *Navigator*. In *Logger mode*, the users may record signal strength information from nearby Wi-Fi access points and upload that data to the *Server*. The *Navigator* is the main mode of operation that allows users to see their current location on top of the floorplan map and see the current location.

Airplace is the predecessor technology for our complete RadioMap-based indoor positioning and navigation platform that operates on top of Google Maps with a big data management Web 2.0 back-end service, coined *Anyplace* [3] (see Figure 5). Anyplace allows entities (i.e., users, companies, organizations, etc.) to realize indoor information management systems, including product search and point of interest (POI) navigation, on top of existing wireless network infrastructure by leveraging rich multi-sensory data available on smartphones.

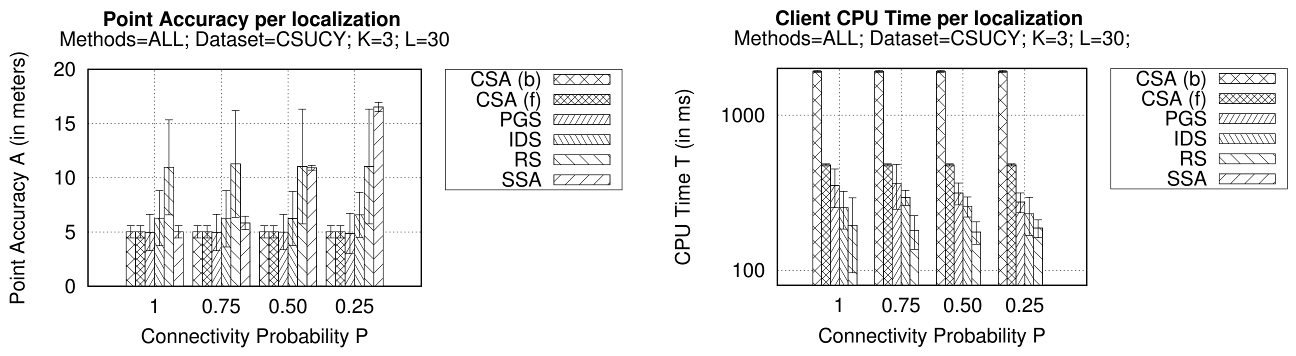


Fig. 6. *Series 1 - Performance Evaluation:* of both non-prefetching and prefetching approaches for indoor localization scenarios in terms of average Point Accuracy A (left) and average CPU Time T (right) while varying the Connectivity Probability P.

B. **Series 1:** *Performance Evaluation*

In the first experimental series, we investigate the performance of *PreLoc* using *PGS*, *IDS*, and *RS* strategies with respect to Point Accuracy (A) and CPU Time (T). We compare all approaches as explained in Section V-A. Note that *CSA* and *SSA* represent the boundaries of performance. The trade-off between accuracy and resource consumption (represented by T) should be clearly demonstrated by *CSA* and *SSA*. The former downloads the whole RadioMap (at a building (b) or floor (f) scale) prior to localizing at the client providing high accuracy, but consumes maximum resources. The latter consumes minimum resources since the client does not download anything, but it provides poor accuracy since the frequent communication with the server-side makes it vulnerable to poorly and intermittently connected networks.

The overall A of the approaches under consideration is relatively high (i.e., ≈5m) due to the fact that fingerprints in the CSUCY dataset are relatively sparse and given that Airplace does not incorporate advanced features introduced later in Anyplace (i.e., ≈2m), e.g., Kalman filters, device diversity, big data, outlier correction, IMU, etc. The results of Figure 6 (left) show that *CSA* approaches do not suffer from intermittent connectivity since the RadioMap is always pre-downloaded a-forehand on the mobile. On the other hand the *SSA* approach is negatively affected by the Connectivity Probability P due to the frequent exchange of information between the client and the server required. The lower the value of P is, the worse the accuracy of *SSA* gets, providing the worst accuracy ($\approx 17m$) in the whole series for $P = 0.25$.

The proposed *PGS* approach of the *PreLoc* framework performs as well as *CSA* and *SSA* approaches for high values of P. The accuracy of *PGS* remains relatively steady and similar to *CSA* as P decreases, but approximately three orders of magnitude better than *SSA*. This shows that *PGS* calculates the correct sequence of fingerprints and therefore it overcomes the intermittent connectivity issues. Moreover, *PGS* provides the best localization accuracy compared to the other two selection techniques, i.e., *IDS* and *RS*, in all cases. Particularly, *PGS* provides 20% better accuracy than *IDS* and $> 50\%$ than *RS*.

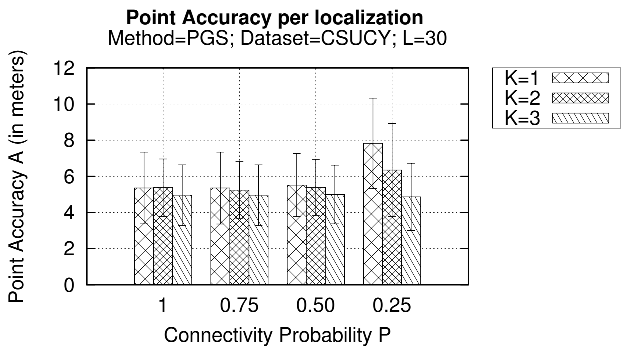
Figure 6 (right) shows that *CSA* as well as the three selection techniques, i.e., *PGS*, *IDS* and *RS*, are not influenced by Connectivity Probability P, providing similar results in all cases. This is due to the fact that all approaches will localize using a fixed number of fingerprints (i.e., the whole RadioMap for *CSA*, K partitions for the selection approaches) regardless of the value of P or the size of the whole dataset. On the other hand, for the *SSA* approach the CPU Time decreases as P decreases due to the fact that intermittent connectivity prohibits the communication between the client and the server resulting in zero CPU effort for localization but also a poor localization accuracy as illustrated in Figure 6 (right).

C. **Series 2:** *Control Experiments*

In the second experimental series, we examine how several parameters influence the behavior and the performance of the proposed *PreLoc* framework with the *PGS* approach. In particular, we initially evaluate the *PGS* accuracy and the CPU Time for localizing 20 consecutive times in four routes of the CSUCY dataset for various K and Connectivity Probability P. We then evaluate the performance of *PGS* in terms of accuracy and CPU Time for a varying number of partitions L and K values.

Figure 7 (left) shows that the accuracy of the proposed *PGS* approach is poor for small values of K, while P decreases. This means that the more intermittently connected a network is the more information (i.e., reference locations for future localizations) is needed to be prefetched. This is the reason why for $K = 3$ the accuracy is good for low P. A general observation is that the accuracy is relatively good for both high P and K. Particularly for $K = 3$ the accuracy provided by the *PGS* is almost fixed. Moreover, the results of Figure 7 (right) show that the CPU Time needed for localization decreases as both K and P decrease. This is due to the fact that smaller K means fewer partial RadioMaps and smaller P means less communication overhead with the server.

In Figure 8, we study the performance of *PGS* in terms of both Point Accuracy A and CPU Time T while varying parameters K and L and fixing $P = 0.5$. Here it is important to realize that the higher the number of partitions in the CSUCY datasets is, the smaller is the size of each individual partition. In other words, if we consider two setups $L = 10$ and $L = 30$ with $K = 3$, then the number of reference locations included in the prefetched partitions in the former setup will be greater than in the latter setup. The results of Figure 8 (left) show that the accuracy is influenced by parameter L, since *PGS*



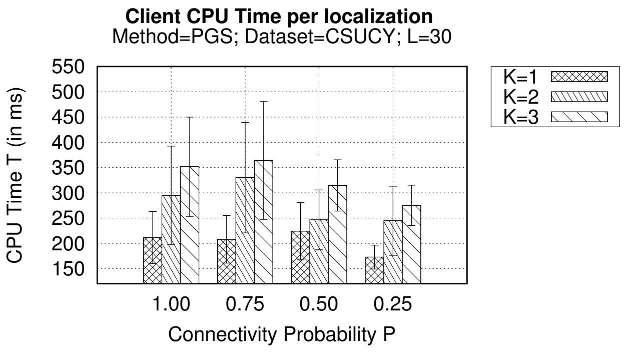
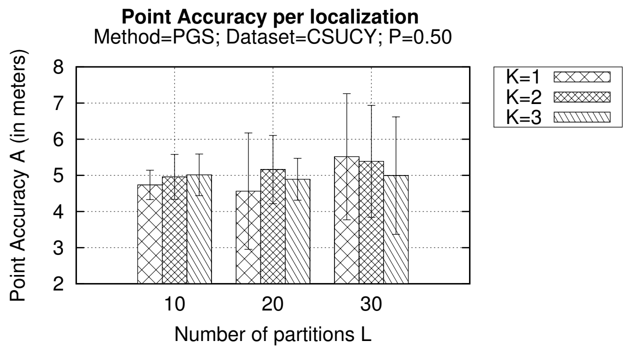


Fig. 7. *Series 2 - Control Experiment:* examining the *PGS* average Point Accuracy A (left) and average CPU Time T (right) while varying the Connectivity Probability P and the selection size parameter K.



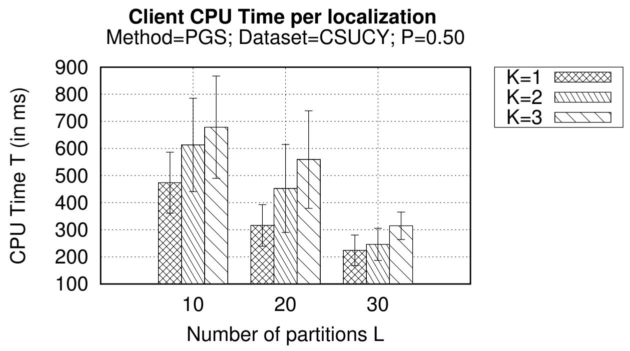


Fig. 8. *Series 2 - Control Experiments:* examining the *PGS* average Point Accuracy A (left) and average CPU Time T (right) while varying the Connectivity Probability P and the number of partitions parameter L.

performs well for small K only when L is small and the performance in terms of accuracy quickly deteriorates when L increases. On the other hand, the accuracy remains relatively good and steady for large values of K under various values of L providing similar results. However, the results of Figure 8 (right) show that the *PGS* performs better when L is high (i.e., partitions with smaller size) and K is small, since less CPU Time is required for localization. Therefore, considering the conclusions drawn from both plots of Figure 8, a good setting would be $L = 30$ and $K = 3$, even though analytically deriving these remains open for future work.

D. Series 3: Energy and Device Diversity

In the last experimental series, we expose the firmness and stability of *PreLoc* framework on a variety of popular Android devices (i.e., Samsung Galaxy S1, HTC Desire and Sony Xperia Z1) in terms of A and T, as well as CPU energy consumption E for varying values of P. The energy consumption of the proposed approach on various devices is measured using PowerTutor[3]. The results in Figure 9 show that *PreLoc* provides similar performance in terms of A in all devices while varying P. The decrease in CPU Time T and Energy E as P decreases is almost linear and it is due to the fact that less data are downloaded and processed. Overall, the behavior of *PreLoc* is consistent under various Android devices and it is, therefore, correct to argue that *PreLoc* is stable.

VI. Conclusion and Future Directions

This paper presents an innovative framework for fine-grained and low-cost indoor navigation with smartphones over intermittently connected Wi-Fi networks, coined Prefetching Localization (*PreLoc*). *PreLoc* operates in three phases: the *pre-processing phase*, where the server partitions the available reference locations of a particular area, the *selection and prefetching phase*, where the server selects a number of partitions to be prefetched, based on historical data of users' past movements on that area, and the *localization phase*, where a disconnected user localizes itself using a well known localization techniques and the available prefetched location data.

We have evaluated our framework using a real prototype developed in Android, as well as real Wi-Fi traces we collected on-campus. Our experimental study reveals that *PreLoc* provides both accurate localization and consumes less resources than existing approaches. In the future, we will further examine the performance of such prefetching approaches, e.g., in terms of latency showing how long the system had to wait as a function of the movement trajectories, in outdoor environments, as well.

[3]PowerTutor, http://powertutor.org

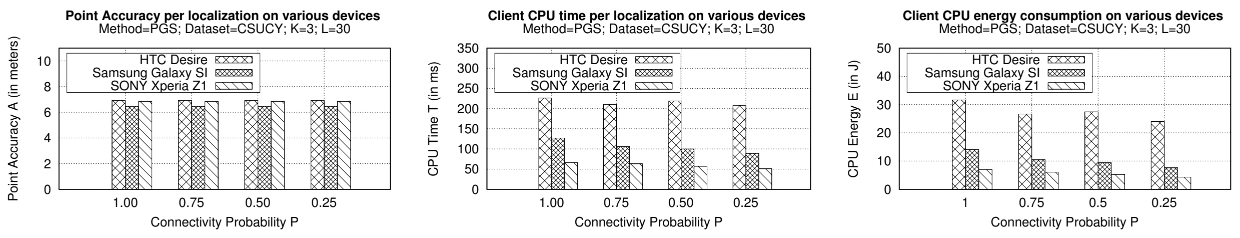


Fig. 9. *Series 3 - Device Diversity:* examining the *PGS* accuracy (left), CPU Time (center) and Energy (right) while varying the Connectivity Probability P.

ACKNOWLEDGMENTS

This work was funded by the Univ. of Cyprus (Startup Grant / Zeinalipour), by MTN Cyprus, EU's COST Action IC903 and IC1304, EU's FP7 MODAP project and EU's FP7 Planetdata NoE. It was also funded in part by the US NSF Award OIA-1028162.

REFERENCES

[1] Y. Gu, A. Lo, and I. Niemegeers, "A survey of indoor positioning systems for wireless personal networks," *IEEE Communications Surveys Tutorials*, vol. 11, no. 1, pp. 13–32, 2009.

[2] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, 2007.

[3] L. Petrou, G. Larkou, C. Laoudias, D. Zeinalipour-Yazti, and C. G. Panayiotou, "Crowdsourced indoor localization and navigation with anyplace," in *Proceedings of the 13th international conference on Information Processing in Sensor Networks*, pp. 331–332, 2014.

[4] D. Lymberopoulos, J. Liu, X. Yang, R. R. Choudhury, ..., C. Laoudias, D. Zeinalipour-Yazti, Y.-K. Tsai, and e. al., "A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned," in *Proceedings of the 14th IEEE/ACM International Symposium on Information Processing in Sensor Networks*, 2015.

[5] Y. Xia and C. K. Yeo, "Mobile internet access over intermittent network connectivity," *J. Netw. Comput. Appl.*, vol. 40, pp. 126–138, 2014.

[6] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw gps data for geographic applications on the web," in *Proceedings of the 17th International Conference on World Wide Web*, pp. 247–256, 2008.

[7] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th International Conference on World Wide Web*, pp. 791–800, 2009.

[8] P. Bahl and V. Padmanabhan, "Radar: an in-building rf-based user location and tracking system," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 775–784, 2000.

[9] B. Li, J. Salter, A. G. Dempster, and C. Rizos, "Indoor positioning techniques based on wireless lan," in *1st International Conference on Wireless Broadband and Ultra Wideband Communications*, pp. 13–16, 2006.

[10] A. Baniukevic, D. Sabonis, C. S. Jensen, and H. Lu, "Improving wi-fi based indoor positioning using bluetooth add-ons," in *Proceedings of the 12th IEEE International Conference on Mobile Data Management*, pp. 246–255, 2011.

[11] A. Baniukevic, C. S. Jensen, and H. Lu, "Hybrid indoor positioning with wi-fi and bluetooth: Architecture and performance," in *Proceedings of the 14th IEEE International Conference on Mobile Data Management*, pp. 207–216, 2013.

[12] P. Prasithsangaree, P. Krishnamurthy, and P.K. Chrysanthis, "On indoor position location with wireless lans," in *Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 2, pp. 720–724, 2002.

[13] G. Larkou, M. Mintzis, P. G. Andreou, A. Konstantinidis, and D. Zeinalipour-Yazti, "Managing big data experiments on smartphones," *Distributed and Parallel Datab.*, pp. 1–32, 2014.

[14] C. Laoudias, D. Zeinalipour-Yazti, and C. Panayiotou, "Crowdsourced indoor localization for diverse devices through radiomap fusion," in *Proceedings of the 2013 International Conference on Indoor Positioning and Indoor Navigation*, pp. 1–7, 2013.

[15] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yazti, "Crowdsourcing with smartphones," *IEEE Internet Computing*, vol. 16, no. 5, pp. 36–44, 2012.

[16] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, and C. G. Panayiotou, "The airplace indoor positioning platform for android smartphones," in *Proceedings of the 13th IEEE International Conference on Mobile Data Management*, pp. 312–315, 2012.

[17] T. M. Lim, C. K. Yeo, F. Lee, and Q. V. Le, "Tmsp: Terminal mobility support protocol," *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 849–863, 2009.

[18] O. Trullols-Cruces, M. Fiore, and J. Barcelo-Ordinas, "Cooperative download in vehicular environments," *IEEE Transactions on Mobile Computing*, vol. 11, no. 4, pp. 663–678, 2012.

[19] Z. Zheng, P. Sinha, and S. Kumar, "Alpha coverage: Bounding the interconnection gap for vehicular internet access," in *Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 2831–2835, 2009.

[20] B. D. Higgins, J. Flinn, T. J. Giuli, B. Noble, C. Peplin, and D. Watson, "Informed mobile prefetching," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, pp. 155–168, 2012.

[21] Z. Zhang, "Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges," *IEEE Communications Surveys Tutorials*, vol. 8, no. 1, pp. 24–37, 2006.

[22] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Efficient routing in intermittently connected mobile networks: The single-copy case," *IEEE/ACM Transactions on Networking*, vol. 16, no. 1, pp. 63–76, 2008.

[23] C. Bouras, A. Konidaris, and D. Kostoulas, "Predictive prefetching on the web and its potential impact in the wide area," *World Wide Web*, vol. 7, no. 2, pp. 143–179, 2004.

[24] J. Wang, "A survey of web caching schemes for the internet," *SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 5, pp. 36–46, 1999.

[25] J. Domenech, B. de la Ossa, J. Sahuquillo, J. A. Gil, and A. Pont, "A taxonomy of web prediction algorithms," *Expert Systems with Applications*, vol. 39, no. 9, pp. 8496–8502, 2012.

[26] J. Grabmeier and A. Rudolph, "Techniques of cluster algorithms in data mining," *Data Mining and Knowledge Discovery*, vol. 6, no. 4, pp. 303–360, 2002.

[27] B. Lent, A. Swami, and J. Widom, "Clustering association rules," in *Proceedings of the 13th IEEE International Conference on Data Engineering*, pp. 220–231, 1997.

[28] P. S. Bradley, U. M. Fayyad, and C. Reina, "Scaling clustering algorithms to large databases," in *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pp. 9–15, 1998.

[29] M. Hazewinkel, "Mahalanobis distance," *Encyclopedia of Mathematics*, Kluwer Academic Publishers, 978-1-55608-010-4, 2002.