Lessons Learned from Bluetooth/Wifi Scanning Deployment in University Campus

Long Vu, Quang Do, Klara Nahrstedt
Department of Computer Science, University of Illinois
Urbana, Illinois, IL61801, USA
{longvu2,quangdo2,klara}@illinois.edu

ABSTRACT

This paper presents the detailed design and implementation of the joint Bluetooth/Wifi scanning framework called UIM[1], which collects both location information and ad hoc contact of the human movement at the University of Illinois campus using Google Android phones. In particular, we present the architecture of UIM and how its sub components interact to obtain the performance reliability as well as conserve phone battery for the prolonged experiment period.

With the movement trace collected by UIM, we first present the findings about number of scanned devices, types of collected devices, and instant cluster size distribution. Then, we study the two graphs formed by the ad hoc trace including connectivity graph and contact graph. We find that the former exhibits a small-world network in structure while the node degree distribution of the latter exhibits an Exponential-Zipf distribution. Finally, we present a novel and efficient algorithm called UIM Clustering to cluster collected wifi access points into clusters and use these clusters to represent locations. Our analysis shows that the distribution of number of locations visited by experiment participants can be fitted by an exponential function.

1. INTRODUCTION

Collecting the real movement trace of mobile users has drawn significant attention from research community since knowledge of human movement is crucial to design network protocols and plan network resources for wireless networks. However, obtaining an accurate human movement trace has remained challenging due to the lack of (1) the portable devices that the experiment participants can carry for a prolonged experiment period, (2) a light-weight, power-efficient scanning protocol that can capture the movement trace and conserve the battery, (3) a device that can be programmed to capture both location information and ad hoc contacts.

In our previous paper [15], we presented a novel joint Bluetooth/Wifi Scanning Framework called UIM, which was deployed on the Google Android phone and carried by faculties/students in University of Illinois campus. Each UIM experiment phone encompasses a Bluetooth scanner and a wifi scanner capturing both Bluetooth MAC addresses and wifi access point MAC addresses in proximity of the phone. The wifi access point MAC addresses then can be used to infer location information while the Bluetooth MAC addresses can be used to infer ad hoc contacts. These two pieces of information are crucial to understand the movement pattern of people. To the best of our knowledge, UIM is the first scanning system capturing both location information and ad hoc contact in one comprehensive movement trace.

In this paper, we present the detail of our design of UIM, which were not presented and explained in [15]. Moreover, in our previous paper [15], we presented the findings from our first round of experiment with 28 participants in March 2010. Meanwhile, in this paper we present the findings from two rounds of experiments including the first round in March 2010 and the second round in April-May 2010. In summary, this paper has the following contributions:

1. We present the detailed design and implementation of UIM including the Bluetooth scanner and Wifi scanner. We discuss the design decision to conserve phone battery and how the sub components inside the scanners interact with each other to obtain the performance reliability.

2. We present the findings from two rounds of experiments about number of scanned devices, device type, and instant cluster size distribution.

3. We characterize the ad hoc graphs including connectivity graph and contact graph. We find that the connectivity graph exhibits the small-world network in structure while the node degree of the contact graph exhibits an Exponential-Zipf distribution.

4. We present a UIM Clustering algorithm to cluster wifi access points into clusters and use these clusters to represent the locations visited by experiment participants. Our analysis shows that the number of location visited by a person fits very well to an exponential function.

This paper is organized as follows. We present the related work in Section 2. Then, we present the detail of UIM design in Section 3. After that, we present the findings of number of scanned devices, device type, instant cluster size distribution in Section 4. Next, we study the graph formed by ad hoc

[1]UIM stands for University of Illinois Movement.

trace in Section 5 and present UIM Clustering algorithm to obtain location from the UIM wifi trace in Section 6. Finally, we conclude the paper in Section 7.

2. RELATED WORK

There have been several efforts in collecting the human movement trace.

The first type of movement traces was collected by GPS-enabled devices carried by experiment participants [14, 10] in which the geographical coordinates of the experiment devices were obtained. However, this trace collection method failed to work correctly if the experiment devices were indoor. More importantly, the collected geographical locations can not be used to infer the connectivity between two geographically closed nodes since there might be obstacles between them. Meanwhile, connectivity is a crucial to evaluate protocols for wireless networks.

The second type of movement traces was collected from WLAN environments where the association between the laptop/PDA and the wifi access points was captured [6, 7]. However, there was a fundamental weakness of these methods since the laptop user did not always turn on the laptop and did not carry it with her all the time. Moreover, a normal laptop user usually turned on her laptop and left it on her desk when doing other things (e.g., had lunch with friends, had meetings with colleagues, or went to exercise at the gym). So, the collected associations of laptops and the wifi access points could be used to understand the wireless usage rather than the detail movement of people.

The third type of movement traces was collected by using portable (experiment) devices such as PDA, iMote, cell phone, which were assigned to participants so that they would carry the devices when they were walking. Due to the limitation of battery and the hardware capability of the experiment devices, only the Bluetooth ad hoc contacts were collected [5, 4, 8, 9, 12, 11]. This method of trace collection captured more accurate movement trace since with high probability, the experiment devices were carried by the participants. However, except [5], all previous works [4, 8, 9, 12, 11] did not capture the location information, which is important to understand the movement of people. For [5], the location was inferred from the cellular base station ID associated with the experiment phone. However, since the transmission range of the cellular base station was ranging from hundred meters (e.g., 500 m) to kilometers (e.g., 30 km), the cellular base station ID did not provide the needed fine granularity. From our observation, the wifi access point could be used to represent the location [1] since a wifi access point usually is associated with a physical building or geographical location. This motivates us in designing UIM to obtain both Bluetooth ad hoc contacts and MAC addresses of wifi access points, and then use wifi MAC addresses to infer more accurate locations (see Section 6).

3. UIM: JOINT BLUETOOTH/WIFI SCANNING FRAMEWORK

This section presents the detailed design of UIM in which we focus on the Bluetooth scanner and Wifi scanner.

3.1 UIM Architecture

As shown in Figure 1(a), UIM has two main components: the database server and the Google Android phone. The former hosts a relational database management system, which accepts and stores the scanning status updates from the experiment phones. The latter has three subcomponents: the Bluetooth scanner, the wifi scanner, and the Status Reporter. The *Bluetooth scanner* periodically (e.g., every 60 seconds) scans the Bluetooth-enabled devices in the phone's proximity [2]. The *wifi scanner* periodically (e.g., every 30 minutes) scans the wifi access points in the phone's proximity. The collected movement trace, including ad hoc trace and wifi trace, is stored at the local disk of the phone. The *Status Reporter* updates the scanning status of the phone (e.g., how the scanning works, how many trace files have been created) to the server via the HTTP connection when the wifi connectivity is available. Due to the battery constraint, we only enable *Status Reporter* at several phones. We find that *Status Reporter* works smoothly if enabled. In the following sections, we present in details the design of Bluetooth and Wifi scanners.

3.2 Bluetooth Scanner

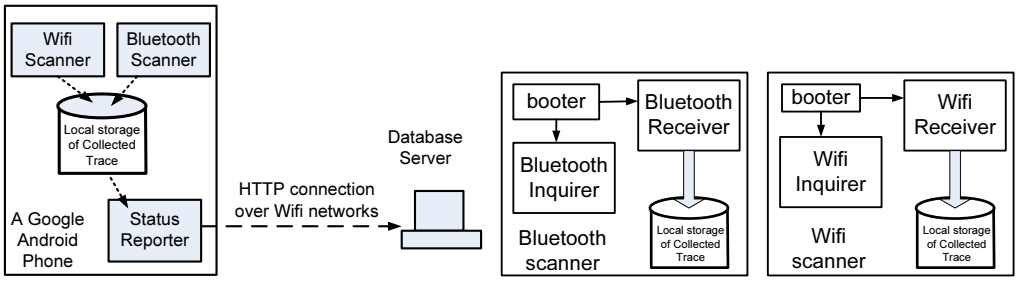
The Bluetooth scanner is shown in Figure 1(b) with three components: booter, Bluetooth inquirer, and Bluetooth receiver. We implement Bluetooth scanner as a background service, anytime the phone restarts, the phone operating system will trigger the booter, which starts the Bluetooth inquirer and Bluetooth receiver. With the booter, UIM obtains the robustness and reliability to run for a prolonged experiment since anytime the phone restarts, the scanner can start its scanning work automatically. The inquirer and receiver work in an asynchronous fashion in which every 60(s) the inquirer uses a request timer to periodically generate a Bluetooth scanning request and send to the system. Then, the inquirer makes the phone discoverable by other experiment phones, goes to sleep, and wakes up for the next request when the timer expires. To conserve battery, we configure the inquirer so that it only generates scanning request from 7AM of a day to 1AM of the next day. During the period [1AM,7AM] of a particular day, the inquirer sleeps. As a result, we can collect most of people movement while saving phone battery for other usages. The receiver, on the other hand, is only triggered to work whenever a Bluetooth scanned result is returned from the phone operating system. Upon receiving the scanned result, the receiver writes the result to the log file with the timestamp and then goes to sleep. Moreover, whenever the phone user triggers the Bluetooth scanning from the phone's GUI, the phone operating system performs a Bluetooth scan and returns result to the Bluetooth receiver. As a result, the Bluetooth receiver *opportunistically* receives Bluetooth scanned results although the inquirer does not trigger the Bluetooth scan. So, the receiver *opportunistically* obtains more scanned result. With this design methodology, the Bluetooth scanner obtains robustness and conserve phone battery.

The scanned results include the Bluetooth MAC addresses and time stamps. In this paper, we use "ad hoc MAC" to denote the scanned MAC addresses. Notice that the ad hoc MAC can be an experiment phone or a scanned device, which is not in the set of experiment phones. So, we use "external ad hoc MAC" to denote a scanned device, which

[2]In this paper we use "participant", "phone", "user", and "experiment phone" interchangeably.

	PMTR	Intel	Cam-City	Infocom	Cam-U	Reality	UIM	Toronto	UCSD	Dartmouth
Environment	Workplace	Corp.	City	Conf.	University Campus					
Duration (day)	19	3	10	3	5	246	55	16	77	114
# of Devices	49	8	36	41	12	97	42	23	273	6648
δ_B(second)	1	120	600	120	120	300	60	120	N/A	N/A
Ad hoc Trace	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Location Trace	No	No	No	No	No	CellID	AP	No	AP	AP
Device Type	PMTR	iMote	iMote	iMote	iMote	Phone	Phone	PDA	PDA	Laptop
# In-contact	11895	1091	8545	22459	4229	54667	40315	2802	195364	4058284
# Ex-device	N/A	92	3586	197	159	N/A	12019	N/A	N/A	N/A
# Ex-contact	N/A	1173	10469	5791	2507	N/A	122091	N/A	N/A	N/A

Table 1: Comparison among collected Bluetooth/Wifi traces



(a) Architecture (b) Bluetooth Scanner (c) Wifi Scanner

Figure 1: UIM Architecture and Sub Components

is not in the set of experiment phones. The trace collected by the Bluetooth scanner is called the "ad hoc trace".

3.3 Wifi Scanner

The Wifi scanner is shown in Figure 1(c) with three components: booter, Wifi inquirer, and Wifi receiver. The design of Wifi scanner is similar to the Bluetooth scanner in which the Wifi inquirer and Wifi receiver are decoupled. Also, the Wifi receiver *opportunistically* receives more wifi scanned results when the phone user triggers the wifi scan functionality from the phone's GUI. However, since the wifi scan consumes much more energy than the Bluetooth scan, we set the scanning period of the Wifi scanner longer.

The scanned results of the Wifi scanner include the MAC addresses of the wifi access points and the corresponding scanning time stamps. In this paper, we use "wifi MAC" to denote the scanned MAC addresses of the wifi access points. The trace collected by the Wifi scanner is called the "wifi trace". There are two reasons the wifi scanning period is set to 30(min). First, in the campus environment, people usually stay in the offices or buildings for a long time period (e.g., a class session is usually 50 minutes). Second, performing wifi scan on the cell phone is energy-consuming.

4. UIM: OVERALL CHARACTERISTICS

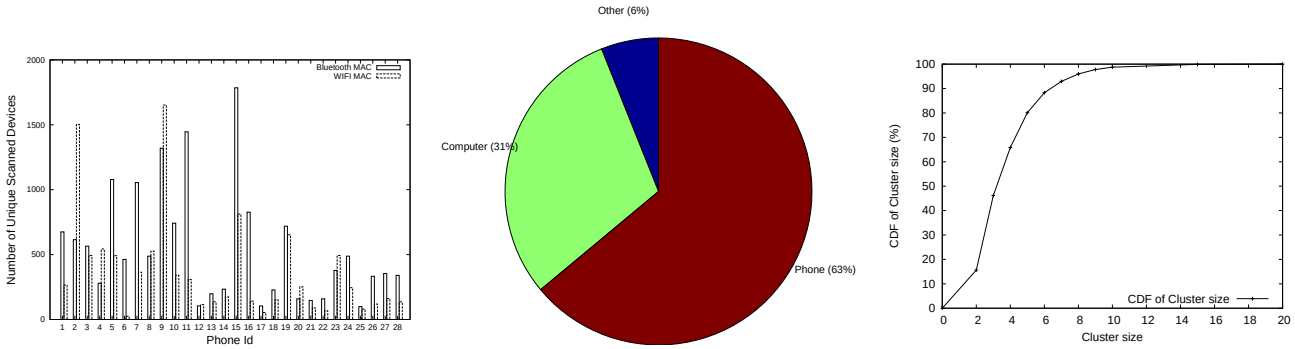
4.1 Comparison of UIM and other traces

We have performed two rounds of experiments with 42 distinct participants. The first round of experiment included 28 participants who carried 28 phones for 19 consecutive days in March 2010 while the second experiment included 16 participants who carried the phones for about 5 weeks from April to May 2010 (here we have 2 participants from the first experiment who continued the second experiment). The participants included faculties, staff, grads, and undergrads as shown in Table 2. The CS faculties, staff, grads usually work inside our department building named Siebel Center. Meanwhile, CS undergrads may take classes in different buildings throughout the university campus. ECE and ABE (e.g., Department of Agricultural and Biological Engineering) grads stay in different buildings from Siebel Center. In this paper, we use D to denote the collected movement trace (including both ad hoc and wifi traces) from 42 phones in our experiment, D_1 to denote the collected movement trace of the first experiment, and D_2 to denote the collected movement trace of the second experiment. So, we have $D = D_1 \cup D_2$.

Table 2 shows the overall statistics of the UIM trace. In this table, for two phones p_1 and p_2, we say that p_1 and p_2 have an "internal contact" if p_1 sees p_2 in its Bluetooth scanned results or vice versa. For a phone p and an external ad hoc MAC address e, we say that p and e have an "external contact" if p sees e in its Bluetooth scanned results.

Overall Characteristics	
# of Internal Devices (participants)	42
Experiment Period (days)	55
Bluetooth Scanning Period (sec, δ_B)	60
Wifi Scanning Period (min, δ_W)	30
# of Internal Contacts	40315
# of External Scanned Devices	12019
# of External Contacts	122091
# of Scanned wifi Access Point MACs	4501
Participants	
# of CS faculties	2
# of CS staff	1
# of CS grads	25
# of CS undergrads	10
# of ECE grads	3
# of ABE grad	1

Table 2: Overall Characteristics of the UIM Trace



(a) Number of Scanned Devices (b) Device Type (c) Instant Cluster Size

Figure 2: Overall Characteristics of the UIM Trace

Table 1 compares the overall characteristics of UIM trace and other previously collected Bluetooth/wifi traces. UIM obtains more detailed and accurate ad hoc contacts since UIM has the highest Bluetooth scanning frequency. For all traces, only Reality [5] used the cellular base station to infer location. As discussed in the Section 2, the cellular base station can not be used for the fine granularity of the physical location. In UIM, we collect the wifi MACs of the wifi access points in the proximity of the phone, which offers more accurate location (see Section 6).

4.2 Number of Scanned Devices

Figure 2(a) shows the total number of unique scanned devices each phone obtained during the first experiment (i.e., the data set is D_1). We see that the numbers are considerably different among phones. Interestingly, many phones obtain more scanned wifi MACs than ad hoc MACs.

4.3 Device Type

Figure 2(b) shows the types of the Bluetooth-enabled devices collected by the UIM system. We see that 63% of scanned devices are phones, 31% of scanned devices are computers, and 6% belong to other types of devices such as headset, GPS, etc. Previous works [4, 8, 9, 12, 11] did not capture the type of the scanned devices. As a result, previous content distribution protocols relying on these collected traces did not distinguish between phones and other Bluetooth-enabled devices in data dissemination. Obviously, for the forwarding of the data message, if the selected forwarder is a mobile node, the forwarder has a higher probability distribute the message to other nodes in the network. In contrast, if the forwarder is a fixed node such as a computer, the forwarding process might be less efficient since the computer (even the laptop computer) may not be carried with the user as much as the phone. The type of the devices is thus useful to design data dissemination protocols, which take into account the device type in selecting the next forwarder of the data message. To the best of our knowledge, we are the first to determine the type of collected devices for the movement traces.

4.4 Instant Cluster

We define the instant cluster C as follows. If two ad hoc MAC M_1 and M_2 appear in one ad hoc scan of the phone p, then (p, M_1, M_2) are in the same instant cluster C. As a result, all ad hoc MACs in one ad hoc scan of the phone p are in the same instant cluster C. For two instant clusters C_1 and C_2, if $C_1 \cap C_2 \neq \emptyset$, then $C_3 = C_1 \cup C_2$ is an instant cluster. So, the definition of instant cluster is transitive.

In order to obtain the instant cluster, we use the ad hoc trace of the data set D_1. For a recorded time stamp t in the ad hoc trace, we consider a time window $[t-45(s), t+45(s)]$ and aggregate all ad hoc scanned results of all experiment phones within this time window into an aggregated record r. Then, we find the instant clusters within each aggregated record r. The 90-second time window is reasonable since we assume that the cluster size remains unchanged during this time window. Figure 2(c) shows that about 90% of clusters in our data set have sizes less than 6, a small cluster size. Since we already aggregate scanned data for 90 (s) when calculating the cluster size, the cluster size of 6 implies that there are not many big clusters in the network. Therefore, protocols in multicasting, content distribution, and congestion control in DTN context [13] may need to take this cluster size distribution into consideration.

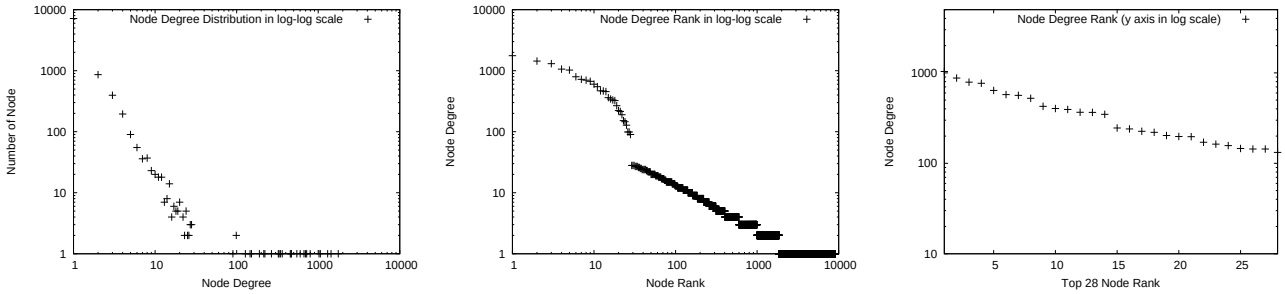
This result has several insights. The cluster size depends on scanning frequency of the Bluetooth scanner, number of experiment phones, and the phone's Bluetooth hardware capability. More importantly, in university campus, grads and faculties usually stay in their research offices. Meanwhile, the Bluetooth scanner can only scan Bluetooth-enabled devices in the range of 10 meters, hence there might not be big clusters. For undergrads, their class sessions can give big clusters of nodes with high probability. However, the scanning range of an experiment phone is about 10 meters while we have a limited number of undergrad participants in the same class session, we may not be able to capture these clusters. Another possibility is there might not be many Bluetooth-enabled devices in the class sessions. However, other trace collection methods face the same challenges.

5. MINING AD HOC GRAPH

We investigate the connectivity graph and contact graph formed by the ad hoc trace of the data set D_1.

5.1 Connectivity Graph

The connectivity graph $G =< V, E >$ is an undirected graph, which is defined as follows. V is the set of nodes, including experiment phones and external ad hoc MACs.



(a) Node Degree Distribution (b) Node Degree Rank (c) Top 28 of Node Rank

Figure 3: Node Degree of the Connectivity Graph $G =< V, E >$

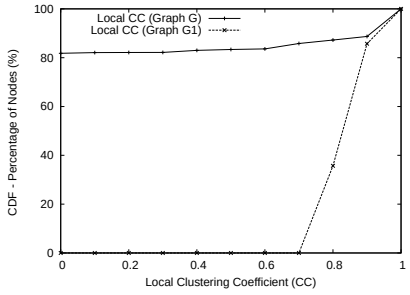


Figure 4: Local Clustering Coefficient Distribution

For a pair of nodes $v_1, v_2 \in V$, if v_1 is a phone and v_2 appears in one scanned result of v_1, then the edge $(v_1, v_2) \in E$. Notice that, if v_1 is a phone and v_2, v_3 exist in one scanned result of v_1, then in our context, $(v_1, v_2) \in E$, $(v_1, v_3) \in E$, but $(v_2, v_3) \notin E$. This definition determines the node degree distribution of G in the following discussion.

Figure 3 shows that the node degree[3] distribution of the graph G follows the Zipf distribution with a heavy-tailed cut-off at the node degree greater than 28. Next, we plot the node rank in terms of node degree in Figure 3(b). This figure shows that for the node rank greater than 28, the node degree follows the Zipf distribution. We then focus on the first 28 nodes in Figure 3(c), which shows that the node degree linearly decreases with respect to the node rank (notice that in Figure 3(c), the y-axis is in log-scale). These 28 nodes are experiment phones, which have a much higher node degree according to the definition of our connectivity graph. We conclude that the node degree distribution of G exhibits an Exponential-Zipf distribution.

We also find that node degree mean of G is 3.26. To further examine the structure of the graph G, we calculate the local clustering coefficient (CC) [16] for all nodes in V. As shown in Figure 4, more than 80% of nodes has $CC = 0$, these nodes are all leaf nodes which have only one neighbor (i.e., the experiment phone). Since 80% of nodes have $CC = 0$, the global CC of the graph is 0.157.

Moreover, we create a connectivity graph $G_1 =< V_1, E_1 >$ of experiment phones and calculate the CC for G_1. Here, V_1 is the set of experiment phones and $|V_1| = 28$. Also, for a pair of nodes $v_1, v_2 \in V_1$, if v_2 appears in one scanned result of v_1, then the edge $(v_1, v_2) \in E_1$. Figure 4 shows that 60% of experiment phones have the local CC greater than 0.8, thus the global CC of G_1 is 0.814, which indicates that the graph formed by phones is highly clustered.

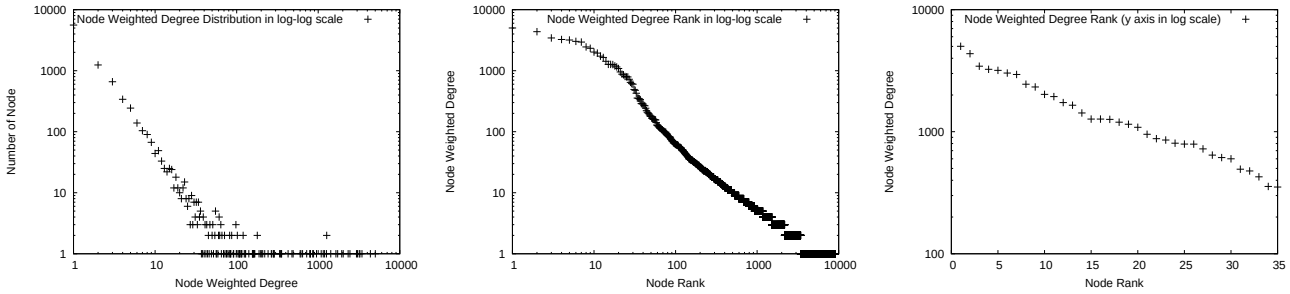
From our analysis, G is a connected graph with 9015 nodes and graph diameter is 4. The low mean of node degree (e.g., 3.26) results from the ad hoc MACs, which are the leaf nodes in the graph with only edges to the experiment phones. From Figure 3(b) we see that the first 50 nodes have degree greater than 25, these nodes form the hubs of G and reduce the graph diameter. Meanwhile, we have only 28 phones, that means the external ad hoc MACs also are hubs in G. This is further confirmed in Figure 4 where the local CC of many phones in G_1 is less than 1, which means there exist cases where the two phones are not connected by a direct edge in G_1. For these cases, the external ad hoc MACs, which are hubs, connect these phones to make G connected and reduce the graph diameter. Besides, although the global CC of the graph G is 0.157, it is considerably greater than the global CC of a random graph with $|V| = 9015$ and mean node degree 3.26 (which is 3.26/9015 = 0.00036). So, we conclude that G exhibits a small-world network in structure.

5.2 Contact Graph

The contact graph $G_C =< V_C, E_C >$ is a "weighted" version of the connectivity graph G. G_C can be obtained from G as follows. For an edge $(v_1, v_2) \in E$, we have a weighted edge $(v_1, v_2)_w \in E_C$, where the weight is the number of contacts between v_1 and v_2 in the ad hoc trace. Notice that we do not present the definition of contact here due to the limited space, however the reader can find the definition of contact from previous studies [15, 4, 8, 9, 12].

For a vertex $v_1 \in V_C$, the weighted degree of v_1 is the sum of weights of edges, which are adjacent to v_1 in G_C. The graph G_C is a connected graph with 9015 nodes. The graph diameter is 4 and the mean of node weighted degree is 9.86. Figure 5(a) shows that the node weighted degree follows a Zipf distribution with a heavy-tailed cut-off. This is confirmed in Figure 5(b) where we plot the rank of nodes in terms of node weighted degree. In this figure, starting from rank 35^{th}, node weighted degree follows very well the Zipf distribution. We then focus on the top 35 node rank in Figure 5(c), which shows that the node weighted degree linearly decreases with respect to node rank. Therefore, we conclude that the weighted node degree distribution exhibits an Exponential-Zipf distribution.

[3]Number of direct neighbors of a node



(a) Node Weighted Degree Distribution (b) Node Weighted Degree Rank (c) Top 35 of Node Rank

Figure 5: Node Weighted Degree of the Contact Graph $G_C =< V_C, E_C >$

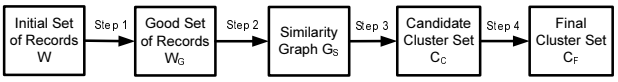


Figure 6: Execution of UIM Clustering algorithm

6. OBTAINING LOCATION

Since the Wifi scanner periodically obtains the set of wifi access point MAC addresses in proximity of the experiment phone, for the entire experiment duration, we have the wifi trace W (e.g., $W \in D$). W consists of multiple records (e.g., $W = \{r_1, r_2, r_3, ..., r_{|W|}\}$). Each record $r_i \in W$ is a set of wifi MAC returned from one wifi scan. For example, the record $r_1 = \{A_1, A_2, A_4, A_{10}\}$, in which A_1, A_2, A_4, A_{10} are wifi MACs. This section presents an efficient algorithm to obtain locations from records of W.

6.1 Motivation and Challenges

In reality, each physical building is usually associated with a set of wifi access points whose MAC addresses can be used as the unique signature to distinguish buildings. If we can obtain the signatures of all buildings visited by the experiment participants, we can construct the spatial-temporal movement patterns of people in the university campus.

There are several challenges in obtaining location from W. First, the wifi wireless scanning range of the phone varies from 100 to 200 meters, depending on various environmental factors such as weather, obstacles. So, for the same physical building, although the phone stays in one fixed position, it may obtain different results for different scans. Fortunately, our experiment shows that these results are usually highly overlapped. Second, there are cases when the phone is in the middle of two adjacent buildings, the scanned result might be partially overlapped with the scanned results obtained when the phone stays in either of the buildings. Fortunately, in reality people stay inside of the buildings more frequently than outside. Thus, wifi access points inside one building appear more often together in the same records of W.

To address these two challenges, we define *location as a unique set of wifi MACs, which appear frequently together in the same records of* W. Next, we present a clustering algorithm called *UIM Clustering Algorithm* to cluster wifi MACs to represent locations using this definition.

6.2 UIM Clustering Algorithm

6.2.1 UIM Clustering Algorithm Overview

Figure 6 shows the execution block diagram of the UIM Clustering algorithm. Particularly, given the initial set of records W, we obtain the sub set of *good* records $W_G \in W$ [4]. Then, we measure the similarity between all pairs of records of W_G and construct a similarity graph G_S, in which each vertex of G_S is a record of W_G. After that, we apply the Star Clustering algorithm [3] to cluster vertices into a set C_C of candidate clusters. Then, candidate clusters are merged based on their similarity measures to obtain the set C_F of final clusters. Each cluster in C_F can be used to represent one location. Table 3 represents the notations used in the UIM Clustering algorithm.

Name	Description
W	The wifi trace $W \in D$, $W = \{r_1, r_2, ..., r_{\|W\|}\}$
W_A	Set of wifi MACs in W,$W_A = \{A_1, A_2, ..., A_{\|W_A\|}\}$
A_i	The i^{th} wifi MAC in W_A
r	A record is a set of wifi MACs in W_A
$\|r\|$	# of unique wifi MACs in r
V_r	The binary bit vector of r, $\|V_r\| = \|W_A\|$
$c(A_i)$	# of records $r \in W$ where $A_i \in r$
$s_{i,j}$	The support value of A_i and A_j
S_r	Support value distribution of wifi MAC pairs of r
m_r	The mean of S_r
f_r	The standard deviation of S_r
F_W	Set of all ratios $\frac{f_r}{m_r}$ of all records $r \in W$
W_G	Set of good records of W, $W_G \in W$
W'_G	Set of binary vectors. E.g., $V_r \in W'_G$ where $r \in W_G$
$T_{p,q}$	Similarity measure of vectors V_p and V_q
θ	The similarity threshold
G_S	The similarity graph: $G_S =< V_S, E_S >$
v_p	One vertex of V_S
C_C	Candidate Cluster Set obtained from G_S
$V^S_{C_i}$	Signature vector of cluster $C_i \in C_C$
C_F	Final Cluster Set obtained from C_C

Table 3: Notations used in UIM Clustering algorithm

6.2.2 Obtaining the Good Set of Records W_G

This Section focuses on the Step 1 in Figure 6. First, we define *a good record as a record that consists of wifi MACs appearing frequently together in the same records of* W. Then, we determine if a record $r \in W$ is a good record as follows: for each pair of wifi MACs $(A_i, A_j) \in r$, we calculate the support value $s_{i,j}$, which represents how frequently

[4] see Section 6.2.2 for definition of good record

the pair (A_i, A_j) appears together in the same records of W:

$$s_{i,j} = \frac{c(A_i, A_j)}{min\{c(A_i), c(A_j)\}} \quad (1)$$

In Equation 1, $c(A_i)$ is the number of records $r' \in W$ in which $A_i \in r'$, the same applies for $c(A_j)$. $c(A_i, A_j)$ is the number of records $r'' \in W$ in which $A_i \in r'', A_j \in r''$. Intuitively, $s_{i,j}$ is similar to the notion of support value in Frequent Item Set Mining literature [2]. For the denominator of Equation 1, we have *min* of $c(A_i)$ and $c(A_j)$ since we are interested in the wifi MAC which appears in less number of records and the association of this wifi MAC with the other one. This *min* value represents the coexistence of the two wifi MACs in the records of W. We have $s_{i,j} \in [0, 1]$ and the greater value of $s_{i,j}$ means the two wifi MACs appear together in the same records of W more frequently.

Let $|r|$ be the number of unique wifi MACs of the record r. We have $\binom{|r|}{2}$ pairs of wifi MACs, thus we have $\binom{|r|}{2}$ support values calculated from the Equation 1 for each pair. These values constitutes a distribution, which we call S_r. Let m_r and f_r be the mean and standard deviation of S_r. Then, we calculate m_r and f_r for each record $r \in W$ using the distribution S_r. If r has only one wifi MAC, $f_r = 0, m_r = 1$. Intuitively, we prefer a greater value of m_r since it means r contains wifi access points that often appear together in the same records of W. Also, we prefer a smaller value of f_r since it means the support values stays in a small range. So, for each record r, we calculate the ratio $\frac{f_r}{m_r}$ to: (1) select good record whose wifi MACs appear together frequently in the same records of W, and (2) remove the bad records, which consists of a mixture wifi MACs of adjacent locations. Let F_W be the set of ratios $\frac{f_r}{m_r}$ of all records of W. We then sort F_W increasingly and create the set W_G from W by applying the Algorithm 1.

Algorithm 1 Obtain W_G from W using F_W, W_A

```
Input: W, F_W, W_A
Output: W_G
BEGIN
W_C = ∅;
for each ratio f_r/m_r ∈ F_W do
  Find the corresponding record r ∈ W;
  M_r = set of wifi MACs of r;
  if |W_C ∪ M_r| > |W_C| then
    W_G = W_G ∪ r;
    if |W_C| == |W_A| then
      return W_G;
    end if
  end if
end for
END
```

The intuition of the Algorithm 1 is as follows. We always prefer records with smaller ratio $\frac{f_r}{m_r}$. Since we need to consider all wifi MACs in W, one record is only useful if adding its wifi MACs to W_C increases the size of W_C; otherwise, the record is filtered out. Doing this, we reduce the number of records and remove most of the noisy data. As a result, the set W_G is good for the clustering algorithm.

6.2.3 Constructing Similarity Graph G_S

This Section focuses on the Step 2 in Figure 6. Given the good set W_G, we map each record $r \in W_G$ into a binary bit vector V_r as follows. If the wifi MAC $A_i \in r$, then the i^{th} bit of the vector V_r is set to 1, $V_r[i] = 1$; otherwise, $V_r[i] = 0$. Notice that, $|V_r| = |W_A|$. Figure 7 shows an example of the binary bit vector.

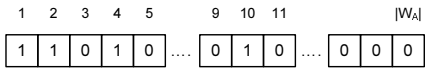


Figure 7: Bit vector V_r of $r = \{A_1, A2, A_4, A_{10}\}$

Let W'_G be the set of binary vectors obtained from all records $r \in W_G$. Then, we use the Tanimoto coefficient (a special form of cosine similarity) to calculate the similarity between a pair of vectors $V_p \in W'_G, V_q \in W'_G$ as follows:

$$T_{p,q} = \frac{V_p \cdot V_q}{||V_p||^2 + ||V_q||^2 - V_p \cdot V_q} \quad (2)$$

Next, we construct the similarity graph $G_S =< V_S, E_S >$, in which each vector $V_p \in W'_G$ is considered a vertex $v_p \in V_S$, so we have: $|V_S| = |W'_G|$. For a pair of vertices $v_p, v_q \in V_S$, the edge (v_p, v_q) exists (i.e., $(v_p, v_q) \in E_S$) if $T_{p,q} \geq \theta$. θ thus determines the topology of G_S and has important impacts on the clustering results (see Section 6.2.6 for detail).

6.2.4 Obtaining Candidate Cluster Set C_C

This Section focuses on the Step 3 in Figure 6. Given the similarity graph G_S, we apply the Star Clustering algorithm [3] to cluster vertices of G_S into clusters. We opt for Star Cluster algorithm since it does not require a pre-defined number of clusters like other clustering algorithms such as partition clustering (e.g., k-means) or hierarchical clustering (e.g., DIANA). Start Clustering thus fits very well to our context since we do not know in advance the number of locations we can obtain from the set of records in W. Applying Star Clustering, we sort the vertices decreasingly according to their node degrees. Then, we scan the sorted list of vertices, for each vertex v_p if v_p is not in any clusters, v_p is considered a center of a new cluster. For each of the neighboring vertex v_q of v_p, if v_q does not belong to any clusters, v_q is included in the cluster centered at v_p. The process continues until all the vertices belong to clusters. We denote this set of clusters *the candidate cluster set* C_C.

6.2.5 Obtaining Final Cluster Set C_F

This Section focuses on the Step 4 in Figure 6. For a cluster $C_i \in C_C$, C_i consists of a set of vertices, each vertex is a binary vector representing a record $r \in W_G$. Let $V^S_{C_i}$ be the signature vector of the cluster C_i. $V^S_{C_i}$ is obtained by applying the OR bitwise operation over all the binary vectors of C_i. Intuitively, the signature vector $V^S_{C_i}$ represents the set of wifi MACs, which belong to the cluster C_i. Thus, the signature vector $V^S_{C_i}$ can be used to uniquely distinguish clusters in C_C. Then, we use the signature vectors to merge cluster $C_1 \in C_C$ into cluster $C_2 \in C_C$ if C_1 is a sub cluster of C_2. Formally, C_1 is merged into C_2 if $V^S_{C_2} = (V^S_{C_1}\ OR\ V^S_{C_2})$. So, we have the final set of clusters C_F, in which each cluster $C_j \in C_F$ can be used to represent one particular location.

6.2.6 Sensitivity of Similarity Threshold θ

To evaluate the impacts of θ, we select the entire set D of 42 participants. For each participant, we select the wifi trace of seven random days and put all records into W, then

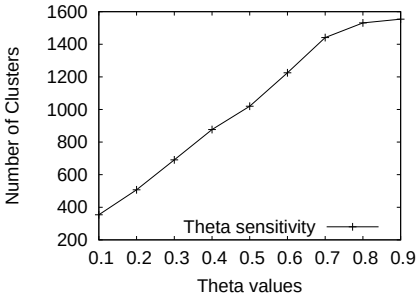


Figure 8: θ sensitivity

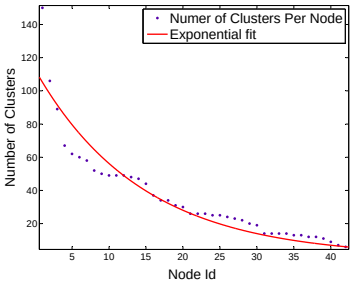


Figure 9: Exponential fitting to number of clusters

we have $|W| = 54564$. Using the ratio $\frac{f_r}{m_r}$ to filter out records, we have $|W_G| = 2315$ good records for the clustering algorithm. In Figure 8, the number of clusters increases nearly linearly when θ increases from 0.1 to 0.9. This result is expected since for greater value of θ, G_S is sparser, so the cluster size is smaller and the number of cluster is greater. To obtain the value of θ for our data set W, we classify all records (including bad and good records) of W into the set of C_F clusters. Each record r is classified to the best matched cluster $C_i \in C_F$ based on the similarity measure between V_r and $V^S_{C_i}$ calculated by Equation 2. Let Δ be the set of all records which are classified into clusters in C_F, notice that $|\Delta| = |W|$. We create a development set W_D by selecting 50 random records from W, in which we manually label the location for each record (e.g., Long's home, Quang's home, Klara's office, etc.). For each value of θ, we perform following steps. For each pair of records $(r_1, r_2) \in W_D$ (i.e., W_D has $\binom{50}{2}$ pairs), we check cluster ids of r_1 and r_2 in Δ and compare these cluster ids with the labeled locations in W_D. Let m_θ be the number of wrong classifications the clustering algorithm makes for all pairs of records in W_D. A wrong classification occurs (1) if r_1 and r_2 have the same labeled location in W_D but they are classified into different clusters in Δ, or (2) if r_1 and r_2 have different labeled locations in W_D but they are classified into the same cluster in Δ. For each wrong classification, we increase m_θ by one. The value of θ with the smallest number of m_θ is desirable. In our data set, $\theta = 0.4$ has the smallest value of m_θ.

Figure 9 shows the number of distinct clusters (i.e., locations) participants visit during 7 days for $\theta = 0.4$. Here, we sort the participants decreasingly according to their number of visited locations. Then, we find that the data in this figure is fitted (by Matlab) very well to the exponential function $y = a \cdot e^{-b \cdot x}$ with $a = 112.8$ and $b = -0.07$. So, we conclude that the number of locations visited by experiment participants can be fitted by an exponential function. In the future, we will use the UIM trace to model the movement pattern and then exploit the model for more efficient data dissemination schemes.

7. CONCLUSION

In this paper, we first present the detailed implementation of the Bluetooth scanner and Wifi scanner of the UIM system. We then present the findings about number of scanned devices, collected device types, and instant cluster size distribution. We also find that the connectivity graph exhibits a small-world network while the node degree distribution of the contact graph exhibits an Exponential-Zipf distribution. Finally, we present the UIM Clustering algorithm to cluster collected wifi access points into distinct locations. We find that the distribution of number of locations visited by experiment participants can be fitted by an exponential function.

8. REFERENCES

[1] Skyhook. http://www.skyhookwireless.com.

[2] *Data Mining: Concepts and Techniques*, pages 225–276. Morgan Kaufmann Publishers, second edition, 2001.

[3] J. Aslam, K. Pelekhov, and D. Rus. Static and dynamic information organization with star clusters. In *Proceedings of the Conference on Information Knowledge Management*, pages 208–217, 1998.

[4] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on the design of opportunistic forwarding algorithms. In *Proceedings of Infocom*, 2006.

[5] N. Eagle and A. (Sandy). Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10:255–268, 2006.

[6] T. Henderson, D. Kotz, and I. Abyzov. The changing usage of a mature campus-wide wireless network. In *Proceedings of Mobicom*, 2004.

[7] W. Hsu, T. Spyropoulos, K. Psounis, and A. Helmy. Modeling time-variant user mobility in wireless mobile networks. In *Proceedings of Infocom*, 2007.

[8] P. Hui, A. Chaintreaum, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *Proceedings of the ACM SIGCOMM workshop on Delay-tolerant networking*, 2005.

[9] J. Leguay, A. Lindgren, J. Scott, T. Friedman, and J. Crowcroft. Opportunistic content distribution in an urban setting. In *Proceedings of CHANTS*, 2006.

[10] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong. Trace ncsu gps, 2009.

[11] E. P. S. Gaito and G. P. Rossi. Opportunistic forwarding in workplaces. In *Proceedings of ACM WOSN*, 2009.

[12] J. Su, A. Chin, A. Popivanova, A. Goel, and E. de Lara. User mobility for opportunistic ad-hoc networking. In *Proceedings of the Sixth IEEE Workshop on Mobile Computing Systems and Applications*, 2004.

[13] N. Thompson, S. Nelson, M. Bakht, T. Abdelzaher, and R. Kravets. Retiring replicants: Congestion control for intermittently-connected networks. In *Proceedings of Infocom*, 2010.

[14] S. van der Spek. *Location Based Services and TeleCartography II (Mapping Pedestrian Movement: Using Tracking Technologies in Koblenz)*, pages 95–118. Springer Berlin Heidelberg, 2009.

[15] L. Vu, K. Nahrstedt, S. Retika, and I. Gupta. Joint bluetooth/wifi scanning framework for characterizing and leveraging people movement on university campus. Technical report, University of Illinois, 2010. http://hdl.handle.net/2142/15465.

[16] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.