MobiShare: Cloud-enabled Opportunistic Content Sharing among Mobile Peers

Kuldeep Yadav, Vinayak Naik, Amarjeet Singh
Indraprastha Institute of Information Technology, New Delhi, India
Email : {kuldeep,naik,amarjeet}@iiitd.ac.in

***Abstract*—In developing countries, smartphones and high bandwidth connections are not yet within the reach of majority of the people. To share multimedia content among themselves, people rely on manual local sharing mechanisms i.e. either using physical exchange of memory card or through short-range wireless technologies, such as Bluetooth. However, this model of content sharing is not scalable in terms of number of users and shared items.**

In this paper, we propose *MobiShare* which is a mobile-cloud based system designed to support searching and sharing of content among pair of mobile peers. The cloud does aggregation of shared content in user's social network and enables scalable content search with real-time computation of expected encounter time between content source and requester. For finding expected encounter time, we build algorithms for (1) Finding mobility profiles of the users using heterogeneous location data services such as Cell ID and WiFi; and (2) Predicting places that will be visited in the future, together with corresponding time spent, using those profiles. We deployed *MobiShare* with 16 volunteers and collected data for 4 weeks. We present evaluation results of the system based on the collected data.

I. Introduction and Motivation

High speed data connection (HSDPA/LTE) exists only with about 30% of total mobile phone users in the world due to reasons such as high data cost and lack of capable smart phones, among others. In the last few years, most of the growth in mobile subscriptions is led by developing countries, such as India, China and African countries, but with limited 3*G* penetration. China has only 14% 3G subscribers of over one billion subscribers[1] whereas India has only has 2% of over 893.8 million subscribers[2]. Further, most phones in developing countries are feature phones with limited capabilities [3], e.g. they lack context sensors such as Global Positioning System (GPS).

Due to limited bandwidth offered by 2*G* networks, it takes time to upload/download large size content (primarily multimedia) and results in significant power consumption. Unavailability of a good Internet connection on mobile phones motivate people to use local sharing mechanisms, such as - (1) Using an intermediary PC/laptop; (2) Physical exchange of memory card; and (3) Short range communication such as Bluetooth. Typically, for all these sharing mechanisms, content seekers manually ask their friends in their social network for the desired content and then seek a rendezvous opportunity (meeting at the same place and time) to exchange the content. Manual checking with friends and need to keep track of all the content makes the current local sharing models difficult to scale for large number of users and content.

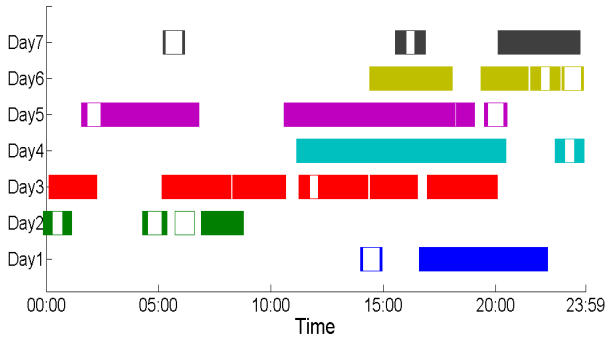
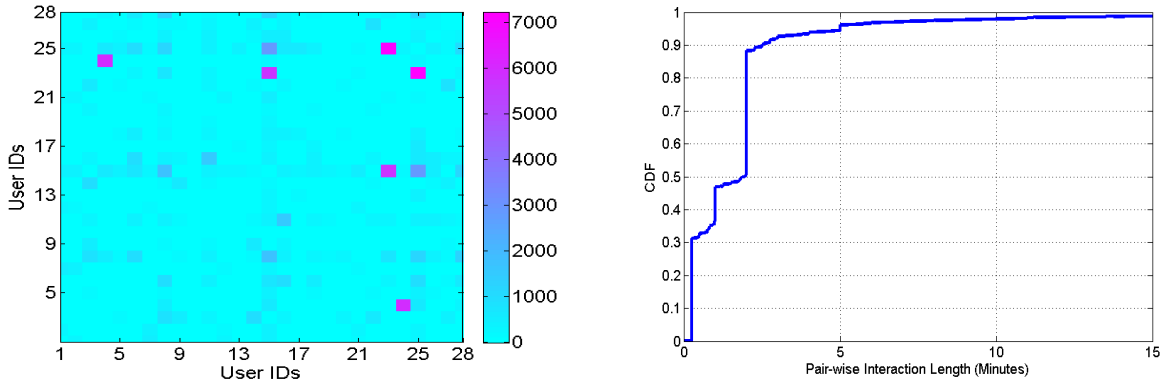


Figure 1: Encounters between a pair of users using WiFi traces in our self collected dataset

There is significant research work for opportunistic sharing of content using mobile phones in two different research communities - 1) Mobile based Delay Tolerant Networks (DTNs) [9], [12]; and 2) Pocket Switched Networks (PSNs) [2], [4]. Current research in mobile based DTNs is mostly limited to designing routing protocols [10], [6], [7] or building distributed content search/sharing protocols [9]. Except few work i.e. [12] and Haggle project [3], much of this work is evaluated in simulations only without realizing system level implementation issues. Distributed content search or sharing protocols assume that a connectivity interface (such as Bluetooth) is always switched on to find opportunities for content exchange with nearby devices which is not realistic in resource-constrained mobile devices. Finally, there is lack of a research work which deals with system level implementation issues (i.e. varying capability of phones) especially in developing world context.

In this paper, we propose *MobiShare* system that facilitates searching and local sharing of content using mobile phones. It is based on a hybrid architecture that uses a central entity i.e. the cloud for storing, aggregating and performing analysis on the information which is uploaded by frontend

[1] http://goo.gl/Ofg00
[2] http://goo.gl/M2l1R
[3] http://www.haggleproject.org/



(a) Heatmap of number of encounters among different pairs of users

(b) CDF of encounter length in minutes among different pairs of the users

Figure 2: Encounter patterns among all users found using Bluetooth data across 3 weeks, 95% of interactions lasts less than 5 minutes.

mobile application. Primarily, the cloud stores information about user's social network, content that is shared by users[4] and location data. To address trust and incentive issues in content sharing, *MobiShare* allows users to search and share content within their social network only.

MobiShare builds on assumption that, there are some frequently occurring (regular) places (such as "Home" & "Workplace") in a person's mobility profile [1], [12] and it is likely that a person will encounter (meet) mostly the same people at these frequently visited places across different days. Figure 2 present encounter patterns found using Bluetooth data using a publicly available dataset from University of Illinois [5]. As shown in Figure 2a, users have high number of encounters with their counterparts during the study but typically their encounter (interaction) length was very short (95% of interactions lasted below 5 minutes).

To show repetitiveness in user encounters, Figure 1 shows encounter pattern among a pair of *MobiShare* users, inferred using our collected WiFi data. It can be clearly seen that the two users spent a lot of time together across different days of the week, e.g. 12:00 to 19:00 across Day 3, 4 and 5, which can be utilized for content sharing. When a user searches for desired content using mobile application, *MobiShare* returns an estimate of time within which content can be delivered. This estimate is predicted using our encounter prediction framework that uses mobility profiles of the users which is built using periodic location updates uploaded from the mobile application to the Cloud.

One of the novelty of *MobiShare* is to build mobility profiles of users using energy-efficient and widely available location sensing frameworks i.e. GSM (Cell ID), WiFi, and Bluetooth which are available on feature phones as well as smartphones. Since, content exchange can only happen when two users are in close proximity, *MobiShare* uses the uploaded location updates and historical mobility profiles to detect physical proximity of the users and correspondingly initiates a neighbor device discovery using Bluetooth or WiFi to find and exchange the desired content. If the content source is not found in the vicinity, communication interface is switched off, until a new rendezvous opportunity arrives. Dynamically switching networking interfaces, minimizes the overall energy consumption of *MobiShare* .

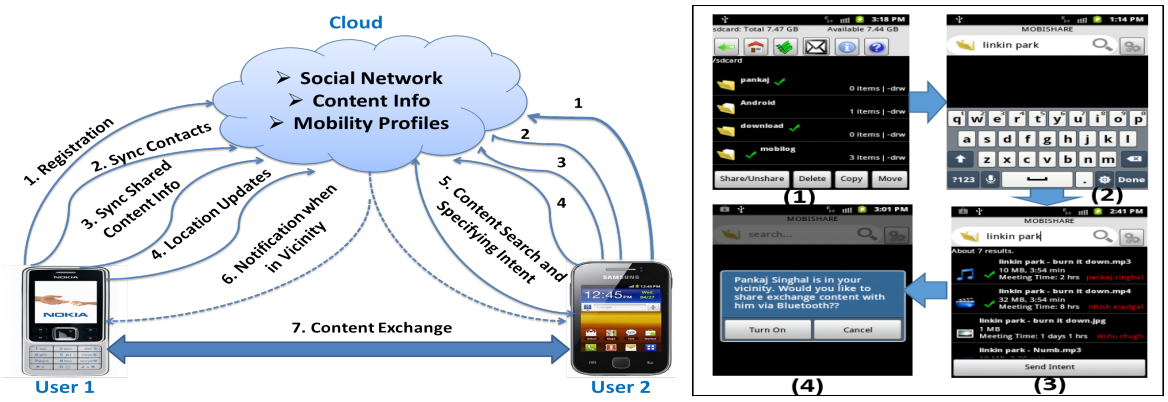
Specifically, the primary contributions of this work are:

1) *MobiShare* system using hybrid architecture for opportunistic content sharing among mobile phones, suited for resource constrained settings of developing countries.
2) Algorithms, using GSM (Cell ID) data only and combined with WiFi data, to learn places which a person visits in a day using raw location data and then accordingly, build spatio-temporal mobility profile of the users and use it to predict future encounters (both time and place). Our evaluation showed that using only Cell ID data, we correctly learn places with nearly 80% accuracy as compared to places learnt using WiFi data. Accuracy improves further i.e. 87% if some days of training can be provided using WiFi data.
3) A proof of concept of *MobiShare* system, built and deployed in mobile phones for 16 voluntary participants in real-world setting for 4 weeks. Collected data[6], together with developed metrics are used to evaluate the accuracy of proposed algorithms.

[4]Here, all the content is assumed to be delay tolerant

[5]http://crawdad.cs.dartmouth.edu/meta.php?name=uiuc/uim

[6]To the best of our knowledge, ours is the first dataset with combined Cell ID and WiFi location information for a developing country setting with other information such as shared content, social network of users etc., we intend to release it publicly after



(a) Mobile-cloud architecture of MobiShare

(b) Mobile application Snapshots

Figure 3: Different aspects of *MobiShare* system - detailed system architecture and snapshots of the mobile application running on a Samsung Galaxy Y

II. System Design and Implementation

MobiShare system has two components - mobile application running on the phones and the cloud service. Information flow in *MobiShare* can be classified as control and data. Control information consists of users' location, meta-data of the files to be shared, users' search queries and content request intent and a notification to share the files. Data information consists of actual multimedia content to be transferred. Different steps for content exchange in *MobiShare* are shown in Figure 3. Small payload size of the control information (utmost few KBs) as compared to actual data, results in quick and low-energy transmission using $2G$ connection. Actual data transfer happens using small range wireless technologies such as WiFi or Bluetooth. We now present a brief overview of different components of *MobiShare* .

A. System Details

Social Network: *MobiShare* forms its own social network for its users, using a combination of phone contacts and Facebook friend list. We bootstrap weights of social links using the previous communication happened between a pair of users (e.g. SMS exchange, call records, Facebook messages). These weights further evolve when users start exchanging content with their friends.

Shared Content: Mobile application provides user an option to select the content that she wants to share. Accordingly, attributes of all the selected files such as name, size, type, genre, album name, artist name are synced to the cloud (Refer Snapshot 1 in Figure 3b).

Content Search: The cloud aggregates global information of available data, across all users, which can be searched in real-time using mobile application. After searching for the content, the cloud also computes the expected delivery time using the mobility profile of users within the social network of the user who is requesting the content (client). All the users are ranked based on computed delivery time to give an estimate of the time by when the client can expect the content to be delivered.

Usage Scenario: We explain the information flow in *MobiShare* using a usage scenario of the system. Let $Alice$, Bob, and $Carol$ be three unique mobile phone users, who are also socially connected with each other. Suppose, $Alice$ wants to have a video "V" but is unaware of who among her friends has this video. Following are the steps, that $Alice$ will follow to get the video "V", using *MobiShare*:

1) $Alice$ opens the *MobiShare* mobile application and searches the video "V" by specifying a set of keywords, similar to a Google search query. (Refer Snapshot 2 in Figure 3)
2) The application requests the cloud to search video "V" in the social network of $Alice$ and finds matches which are lexically similar to the search query.
3) Assuming that the video "V" is available with both Bob and $Carol$, the cloud computes the expected delivery time of $Alice$ with Bob, and $Carol$ by looking at mobility profiles of all the three users, ranks the search results based on delivery time, and returns the sorted results to the mobile application.
4) From the returned results, $Alice$ selects the user (s), say Bob from whom she wants to fetch the video (Refer Snapshot 3 in Figure 3).
5) The mobile application informs the cloud about the *intent* that $Alice$ wants to receive the "V" from Bob.
6) The cloud thereafter tracks locations of $Alice$ and Bob to infer the time when they come in physical proximity and then send notifications to both users (Refer Snapshot 4 in Figure 3)
7) After reception of the notifications, both the devices starts neighbor discovery to find each other and ex-

Data	Total	Average (per User)
Number of GSM location updates	308680	19292
Number of WiFi location updates	31873	1992
Number of shared files	1766	110
Content request intents	250	16
Number of mobile contacts	2656	166
Number of Facebook friends	5264	329

Table I: Descriptive statistics about the data set collected as part of *MobiShare* deployment with 16 participants

change the video "V" if the discovery is successful. If the discovery is unsuccessful, the short range communication is switched off and then periodically turned back on to recheck and save on overall energy consumption.

B. Data Collection

We deployed *MobiShare* with 16 participants in New Delhi, India which includes graduate and undergraduate students and they have used it for 4 weeks. Participants had subscriptions from five different cellular operators. *MobiShare* imported Facebook friend list and mobile contacts of the participants to infer their social network. To build the mobility profile of the users, mobile application scans and logs GSM information every 2 minutes and visible WiFi access points (APs) every 15 minutes. Bluetooth is turned on demand whenever there is a content exchange opportunity. GSM information consists of timestamp, MCC (Mobile Country Code), MNC (Mobile Network Code), LAC (Location Area Code), and Cell ID (Identifier of base station to which phone is currently connected). The WiFi information consists of current timestamp, BSSID, and SSID of the visible APs. Large intervals for WiFi scans are set as it consumes significantly higher energy as compared to GSM [17]. Mobile application provides the user with an option to automatically sync collected information to the cloud at any interval between five to thirty minutes. Our participants shared total 1766 files (mean : 110) with their social network and there were total 250 intents were registered to get content from each other. Out of 16 users, 3 users had very limited or no WiFi data. For further analysis, we considered data of only 13 users who had minimum three weeks of WiFi data.

III. User Mobility Profiling

User's *Mobility Profile* is defined as a spatio-temporal representation of her mobility, i.e, different places visited by her, with their arrival and departure time for every day. A place is defined as a location, where the user stays for a significant amount of time, e.g., "Home" and "Workplace". Burbey et al [19] considered a location as a place if the user has spent more than 10 minutes at that location. For building mobility profile, one of first step and challenging task is to find different places that a person visited using raw location data and then use this place information to find arrival and departure time specific to those places.

There are several location interfaces available on a mobile phone which can be used to build mobility such as GPS, WiFi, GSM, and Bluetooth etc. Each of these interfaces differ in terms of accuracy, availability, and power consumption. Although, GPS provides high accuracy, it also consumes high power when used continuously for a long time period [16], also it has very limited availability because feature phones usually does not GPS chip and GPS can not provide location in indoor environment. From the measurements done across different mobile operating systems and devices, Friedman et al [13] found that scanning modes of WiFi and Bluetooth also consume significant power and quickly drain the battery in continuous location sensing applications.

A. WiFi based Mobility Profiling

MobiShare collects WiFi records consisting of nearby WiFi APs, scanned by the mobile client every 15 minutes, together with corresponding timestamp. Collected WiFi records are periodically sent to the cloud, as described in Section II-B. In the cloud, daily WiFi records are analyzed to build users' mobility profiles. Each WiFi record contains a vector of visible WiFi MAC addresses, SSIDs, BSSIDs, and timestamps. For building mobility profiles, *MobiShare* first finds different places visited by a user using WiFi records, i.e., clusters of observed WiFi APs w.r.t. physical places. The primary assumption in this process is that the user will see different set of WiFi APs at different places due to limited transmission range of the WiFi APs.

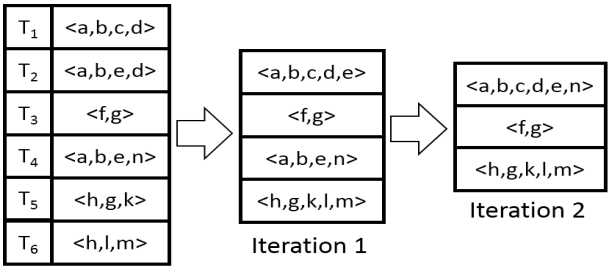


Figure 4: Representation of different iterations in WiFi APs Clustering Algorithm

Consecutive WiFi scans at the same physical place should result in same or nearly same WiFi APs. Using time-ordered WiFi records as shown in Figure 4, *MobiShare* clusters WiFi APs, by joining different WiFi records, if they have at least one common APs between them. The joining process continues till there is no further joining possible among the WiFi APs as shown in Figure 4. The process will result in set of WiFi clusters W, containing non-overlapping WiFi APs, where each cluster $W_i \in W$ is likely to correspond to a different place in the user's mobility profile.

Place IDs	Monday	Tuesday	Sunday
P_1	00:49-10:28 21:39-23:59	00:01-07:53 19:27-23:59	00:53-22:12 22:57-23:59
P_2	11:08-18:38	08:17-19:24	
P_3	19:10-21:11		

Table II: A spatiotemporal mobility profile generated by the *MobiShare* cloud system from a week's data of user X using WiFi records, there are three different places visited by this user where much of the time is spent on places P_1 and P_2

From our collected data, we observed that sometimes WiFi scan results in *partial-scan*, wherein adjacent WiFi records at the same place do not contain any common WiFi APs. For instance, WiFi records shown in Table 4 at time T_3 belong to the same physical place as records at T_1, T_2 and T_4 but T_3 does not share any common APs with them. To solve this *partial-scan* problem, we maintain an association of WiFi APs from previous days. If a pair of APs, say $< a, b >$, were in the same cluster on day i, we use that association for subsequent days and always put a and b into the same cluster by doing another iterative join operation on resulting clusters W.

We involved participants at the beginning to human-label each cluster in W. We selected six participants, presented them with the information about clusters produced, and asked them to provide a label for each of the clusters. Information presented to the participants consisted of corresponding AP's names, arrival time, departure time, and time spent in the respective clusters from their own data only. Participants were advised to put human readable and consistent names for all the places across different days. As an example, they put names as "Home", "Workplace", "Market", "Library", or "Unknown" (in case, they do not remember). To speed up the process and to reduce the overload on the participants, we asked them to label clusters for three weeks of data only.

Using human-labeled WiFi clusters, we found that all the clusters in W represented distinct places in a user's mobility profile, thus confirming the accuracy of the clustering approach used. After accurately finding clusters and given time ordered time ordered WiFi records for a day, we build the mobility profile of that person. If a person spends less than 15 minutes at a place, we do not include it into that user's mobility profile. A weekly mobility profile generated for a user X, using our approach, is shown in Table II.

While mobility profile built using WiFi records is accurate and correctly depicts all the places visited by a user, repetitive WiFi scans are power hungry and increase the battery consumption of a phone. In most of developing countries, even WiFi has very limited availability due to lack of infrastructure. In our collected data, WiFi was available only about 60% of the time as compared to GSM. We collected WiFi information to estimate the ground truth of user mobility profile and evaluate the accuracy of the proposed GSM based approaches.

Cell ID	Start Time	End Time
C_1	t_1	t_2
C_2	t_2	t_4
C_1	t_4	t_5

Table III: Representation of time ordered cell records

B. Cell ID (GSM) based Mobility Profiling :

GSM information (MCC, MNC, LAC, and Cell ID), collected by *MobiShare* , is used to compute *cell records*, which consist of Cell ID and time spent by user in that Cell ID in a continuous interval. As an example, if a user's movement pattern in terms of Cell IDs is $\{C_1,C_2,C_2,C_1,C_1\}$ at time $\{t_1,t_2,t_3,t_4,t_5\}$ respectively and further, $\forall_{i\epsilon\{1,5\}} t_{i+1} - t_i \leq \alpha$, it will result in a cell record as shown in Table III. *MobiShare* uses parameter α to take care of missing Cell IDs due to switching off of phones, unavailability of the network, and loss of location updates. Value of α used is 4 minutes, twice that of sampling interval of GSM information.

Finding distinct places using only Cell ID information has several challenges. Previous work [5] has shown that even if a user stays at the same place, the Cell ID may change due to various reasons such as network load, small time signal fading, and inter-network ($2G$ to $3G$ or vice versa) handoff. This change in Cell ID at the same place is called as an "oscillating effect". We also observed oscillating effects in our collected data that motivated the algorithms, discussed next, that intelligently use Cell ID information to accurately represent spatially disjoint places.

1) LAC-based Cell ID Clustering Algorithm (LCA): Cellular network assigns a group of cell base station in a location area with the same identifier, known as Location Area Code (LAC) [18]. Distinct places using Cell ID information can be found out by using LAC with an assumption that each place visited by user will fall under different LAC. From our collected data, we extract daily cell records and and use them to create different clusters (C_l), each having multiple Cell IDs within the same LAC. We compare the accuracy of LAC based clustering with ground truth in Section III-B4.

2) Graph-based Cell ID Clustering Algorithm (GCA): Assuming that $\{C_1,C_2,C_3,....C_k\}$ are the distinct time-ordered Cell IDs observed in a day, we build an undirected graph, called as *movement graph*, $G(V, E)$ where $\forall_{i\in\{1,k\}} C_i \in V$ and there exist an edge $e(C_i, C_j)$ between C_i and C_j, if both of the following conditions are satisfied:

1) C_i and C_j are contiguous in time ordered cell records
2) Time difference between start time of C_j and end time of C_i is less than α.

As an example in Table III, C_1 and C_2 occurred contiguously and t_2-$t_2 \leq \alpha$, so there will be an edge between C_1 and C_2 in the corresponding movement graph of the user. Multiple edges between C_i and C_j are merged into

a single edge with weight equal to the number of edges between C_i and C_j. α ensures that an edge occurs only across neighboring (in time) Cell IDs and other cell records, that may be neighboring but with a high time difference between them due to reasons such as switching off of the phone, unavailability of the network, and loss of location updates, are pruned. An example of movement graph created from user X's data is shown in Figure 5.

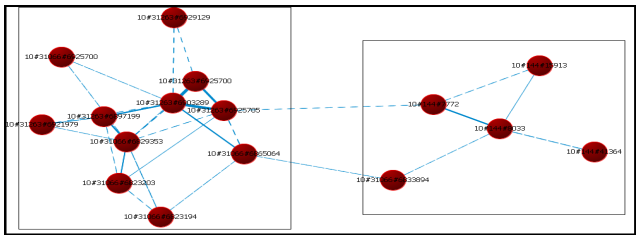


Figure 5: A snapshot of a user's movement profile and corresponding places identified (within the square box)

As illustrated in the movement profile in Figure 5, for each of the two places (represented using different rectangles) visited in a day by X, several Cell IDs are observed with multiple fluctuations amongst them. To cluster Cell IDs, accounting for the oscillating effect, into different places visited by the user, we propose a three phase algorithm as described in Algorithm 1. *Cell Clustering Algorithm* takes movement graph as an input and produces Cell ID clusters as an output, where each cluster will represent a different place. We define a parameter called an *oscillation parameter* η, which measures the number of fluctuations between a pair of Cell IDs in a day. As an example, for Table III, η will be 2 between Cell ID C_1 and C_2 and vice versa. In case of a movement graph, η between C_i and C_j is same as edge weight between C_i and C_j.

In the first phase of *Cell Clustering Algorithm* , we start with initializing cluster set CG to ϕ. All edges in the movement graph $G(V, E)$ are then sorted according to their weights in descending order. After sorting, *Cell Clustering Algorithm* selects one edge at a time and checks if at least one of its vertex is already in one of the cluster sets (say $CG_i \in CG$). If it is found, both the vertices of that edge are merged into CG_i. Otherwise, a new cluster is created containing only those two vertices and added to CG. We define a variant of earlier oscillation parameter, called η', that measures the number of transitions from a Cell ID to any other Cell ID in the movement graph in a day. η' will be essentially equal to the degree of a vertex. In the second phase, we consider each vertex within the set of clusters (CG) and if the degree of vertex (say v) is higher or equal to η', all the neighboring vertices of v are also added to the respective cluster, if they are not already included.

To create distinct and non-overlapping clusters, in the third phase, we combine all the clusters in CG, which have

Algorithm 1: Pseudocode of *Cell Clustering Algorithm*

Algorithm: Cell Clustering Algorithm
Input: Movement Graph $G(V, E)$ where V is set of vertices and E is the set of edges
Output: Set of Cell ID Clusters CG
begin
  Rank all the edges in E into decreasing order of their weight;
  $CG = \phi$;
  while $(\forall e_k \in E)$ *AND* $w(e_k) \geq \eta$ **do**
    if $v_i \in CG_j$ *where* $v_i \in e_k$, $i \in (1, 2)$, $CG_j \in CG$ **then**
      $CG_j = CG_j \cup v_{k1} \cup v_{k2}$;
    else
      Create new cluster $CG_n = v_{k1} \cup v_{k2}$ and add it to CG ;
  while $(\forall v_j \in CG_k)$ *where* $CG_k \in CG$ **do**
    if $(degree(v_j) \geq \eta')$ **then**
      $CG_k = CG_k \cup neighbors(v_j)$;
  $CG' = \phi$;
  while $(\forall CG_i \in CG)$ **do**
    $isExist = false$;
    while $(\forall CG_j \in CG')$ **do**
      if $(CG_i \cap CG_j) \neq \phi$ **then**
        $CG_i = CG_i \cup CG_j$; $isExist = true$; break ;
    if $\neg(isExist)$ **then**
      Add CG_i to CG' ;
  while $(\forall v_i \in V)$ **do**
    if $(v_i \notin \exists CG_k)$ *where* $CG_k \in CG'$ **then**
      Create new cluster $c_n = v_i$ and add it to CG' ;
  return CG' ;

a common vertex, among themselves to produce a new set of clusters CG' that does not have any common vertices across different clusters. Finally, for each left over vertices, which do not belong to any of the clusters in CG', we create a separate cluster and add those clusters to CG'. Now, each cluster in CG' represent a different place visited by the user in a day and subsequently can be used to build user's mobility profile for the day.

3) WiFi Trained Cell ID Clustering (WTCA): GCA inherently assumes that different places in a user's profile will have non-overlapping sets of Cell IDs. As a result, for a person visiting distinct places that are in close proximity, e.g. a student staying in a dorm that is close to the academic building, GCA will merge the two different places if a common Cell ID is observed at each of the two places. This merging effect of GCA is also observed in our collected data as some of our users also live in campus residence.

While such geographically close places may have overlapping Cell IDs, it is unlikely that they will have overlapping WiFi APs. Further, not all Cell IDs will overlap across the two distinct places. We use these two insights to extend GCA by training it with WiFi based Cell ID clustering. For training purpose, we use the WiFi mobility profile to determine corresponding Cell ID clusters, accounting for arrival and

departure time at a place. If we have an initial WiFi training data for d days, we create WiFi trained Cell ID clusters observed for each day, say $CW = CW_1, CW_2, ...CW_d$.

A Cell ID is said to be conflicting if it belongs to two different clusters within the same day. Such conflicting Cell IDs essentially belong to two different places and hence can not be relied upon when performing clustering based only upon Cell IDs. We create a separate conflicting set, C_C, that contains all such conflicting Cell IDs which exist in CW. To cluster the remaining non-conflicting Cell IDs in CW into unique places, we create a support metric $s(C_i, C_j)$ for every non-conflicting Cell ID pair $C_i, C_j \in CW$ as $s(C_i, C_j) = \frac{O(C_i,C_j)}{min(O(C_i),O(C_j))}$, where $O(C_i, C_j)$ denotes the number of joint occurrences (in days) of C_i and C_j within the same cluster and $O(C_i)$ denotes all the occurrences (in days) of C_i, irrespective of whether C_j was in the same cluster as C_i or not. Two Cell IDs C_i and C_j are strongly connected, and are likely to be in the same cluster, if $s(C_i, C_j) \geq \gamma$, where γ is system defined threshold.

Specifically, daily Cell ID clusters made using GCA (CG), are refined to remove the merging effect using WiFi based learning as follows:

1) For each cluster $CG_i \in CG$, remove the Cell IDs that overlap with the conflicting set C_C, and insert them into a separate cluster called C_S. Correspondingly, CG is modified to a cluster set CG' for which Cell IDs across all the clusters are non-overlapping.
2) Separately, for each cluster $CG'_i \in CG'$, if it was affected in the previous step then it is taken out from CG' and all its Cell IDs are added into CG_n. Thereafter, Algorithm 2 is used to return one or more strongly connected clusters in CG_n, called SC, for the affected cluster CG'_i.
3) Separately for each of the strongly connected clusters, add back the corresponding conlicting Cell ID from C_S to each of its components, to create the modified strongly connected clusters SC'.
4) Final cluster set CG is obtained by combining clusters in SC' and CG'.

Using the WiFi training data, WTCA algorithm corrects the merging error of GCA algorithm and subsequently, forms new set of Cell ID clusters which will result in more accurate mobility profile compared to GCA. Table IV shows mobility profile for user X which is build by *MobiShare* cloud system from heterogeneous location sources such as Cell ID, WiFi and Bluetooth. We now evaluate the accuracy of all clustering algorithms with the WiFi based ground truth.

4) Evaluation: Using WiFi mobility profiles, we associate Cell IDs with actual physical places which is used as ground truth for evaluation of Cell ID based clustering approaches, GCA and LCA. For GCA, we empirically found η and η' to be equal to 3 and used it for performing all experiments related to GCA.

Algorithm 2: Pseudocode of Strongly Connected Clustering Algorithm

```
1  Algorithm: Strongly Connected Clusters Algorithm
   Input: CG_i is a cluster of Cell IDs
   Output: Strongly connected clusters set SC
2  begin
3      SC = φ ;
4      while (∀C_j ∈ CG_i) do
5          while (∀C_k ∈ CG_i) do
6              if s(C_j, C_k) > γ then
7                  if (C_j ∈ SC_m OR C_k ∈ SC_m where
                   SC_m ∈ SC) then
8                      SC_m = SC_m ∪ C_j ∪ C_k ;
9                  else
10                     Create a new cluster with (C_j, C_k) and add
                       it to SC ;
11     return SC ;
```

Place IDs	Time	Cell IDs	WiFi APs	Bluetooth APs
P_1	00:49-9:30	C_1, C_2, C_3	W_1, W_2, W_3	BT_1, BT_2
P_2	9:55-6:34	C_5, C_6, C_7	W_6, W_8, W_9	BT_6, BT_7, BT_8
P_3	8:45-11:59	C_1, C_2	W_1, W_2, W_3	BT_1

Table IV: A spatiotemporal mobility profile generated by the *MobiShare* cloud system by aggregating heterogeneous location data of user X for a day, it does not contain places where user spend less than 15 minutes of time

We define Cell ID clusters made using WiFi, GCA and LCA as CW, CG and CL respectively. We further define a pair-wise comparison metrics, called *Correct Pair* for our evaluation. A Cell ID pair (i.e. C_i and C_j) is counted as *Correct Pair*, if their occurrence within the same or across different clusters in CW is reflected accordingly in the Cell ID based approach evaluated. Correspondingly, to evaluate GCA, we first calculate the % of Correct Pairs, out of total pairs, of Cell IDs across CW and CG, and then take an average across different days to compute the final accuracy of GCA. Similar process is followed w.r.t CW and CL and accuracy of LCA is computed.

As shown in Figure 7, GCA produces 80.29% correct pairs (on an average across all users and all days) while LCA produces 70.82% correct pairs (on an average). For some users, LCA performs slightly better or equivalent to GCA, which is essentially when LCA's underline assumption becomes true, i.e. a person visits places that are in different LAC areas. However, as one can observe in Figure 7, this condition is true for only a few users. Errors in GCA occurred since it mistakenly merged places that were geographically close (as discussed earlier).

While calculating correct pairs using WTCA, we ignored Cell ID pairs containing conflicting Cell IDs. For calculating the strongly connected components, we empirically found the γ value of 0.5 to give maximum possible accuracy. We also evaluated the effect of number of days of WiFi training

(d) on accuracy of WTCA. Figure 6 shows the average number of correct pairs produced by WTCA across all users with different training days. As can be observed, after $d = 8$, accuracy remains nearly same for subsequent days.

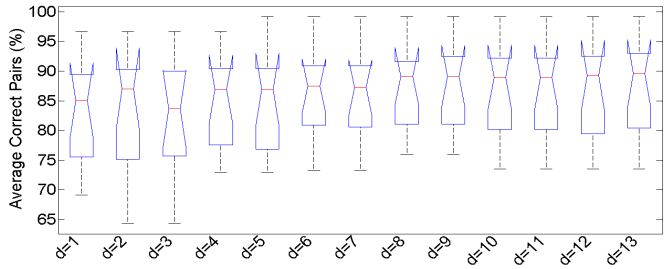


Figure 6: Average accuracy comparison of WTCA across all users with different number of training days i.e d

With $d = 8$, we compared the accuracy of WTCA with LCA and GCA, as shown in Figure 7. WTCA either equals or improves % of correct pairs across all users as compared to GCA or LCA. On an average across all users and days, WTCA produces 87.30% correct pairs as compared to GCA which produces 80.29% correct pairs. WTCA improves upon the overall accuracy of clustering, when compared to GCA, since it can split merged places and put them into different clusters, using the training data.

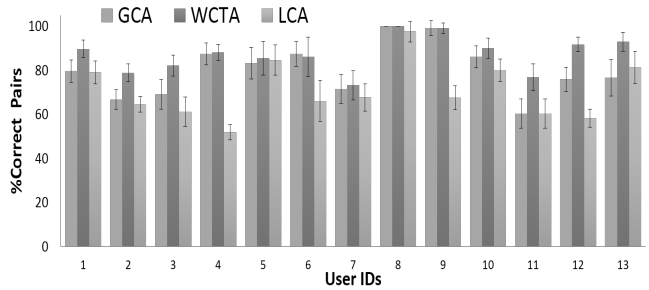


Figure 7: Comparison of GCA,WTCA, and LCA w.r.t. ground truth. On an average, WTCA produce more correct pairs (87.30%) than GCA (80.29%) and LCA (70.82%)

IV. Encounter Prediction of Users

When a user makes a query for a content, *MobiShare* provides her with an expected content delivery time. This content delivery time is the difference between the time when the file search is performed and the time when the user is expected to meet another user, who has the desired content. For an actual content transfer to successfully happen across the two users, we need to predict the common place where they are likely to meet (i.e. *encounter place*) and the time spent at the common place (i.e. *encounter time*) together. We call predicting *encounter place* and *encounter time* together as predicting *encounters*.

A. Finding Overlapping Places

To find encounter places and corresponding encounter time, we need to build a mapping between places visited by the two users. For instance, if a user U_i visits a place P_i from 9:30 to 17:30 and another user U_j visits a place P_j from 11:00 to 18:00 and it can be established that P_i are P_j correspond to the same place, then for U_i and U_j, encounter place is P_i (=P_j) and encounter time is 11:00 to 17:30.

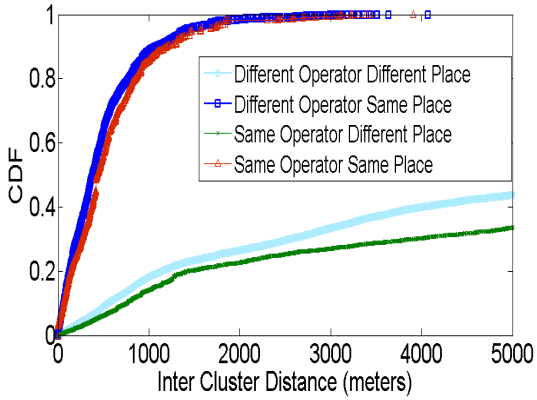


Figure 8: CDF of ICD with same and different places across different operators discovered using WiFi data

In *MobiShare*, as mobility profiles are built using heterogeneous location data sources and the mobility profile may contain places that are unique to a user, finding an encounter place can be a challenging task. In case of WiFi based mobility profiles, presence of one or more common WiFi APs across the two places can potentially be taken as a simple way to infer them as an encounter place. For GSM based mobility profile, when two users belong to the same operator, overlapping Cell IDs of respective places can be used to infer encounter places. However, if the two users belong to different operators, there will be no overlap between Cell IDs (since each operator has a different range of Cell IDs), thus complicating the task of finding encounters.

We resolve this problem by taking into account Cell ID geo-coordinates, that are fetched using Google's GeoLocation API[7]. We define a metric *Inter Cluster Distance* (ICD) which measures the distance between centroid of two clusters, calculated by taking average of geo-coordinates of all Cell IDs that belong to the cluster. To differentiate between two places, using ICD, we need to estimate a threshold distance s.t. ICD $>$ threshold will imply different places. We used our collected data to estimate this threshold. We selected pairs of Cell ID clusters, that belong to two different users and calculated ICD between them. To find if a given pair of Cell ID clusters belong to the same place, we generate ground truth using WiFi APs. Figure 8, displaying

[7]http://code.google.com/p/gears/wiki/GeolocationAPI

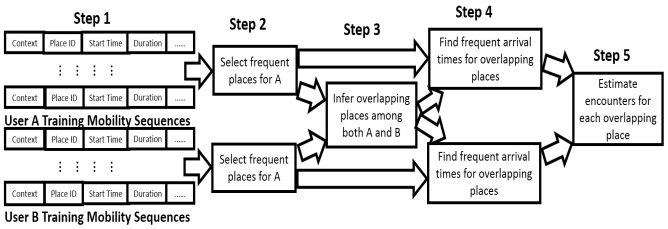


Figure 9: Step by step execution of proposed encounter prediction engine

cumulative distribution function (CDF) of ICD for different settings, shows that a pair of Cell ID clusters that represent the same place across two users will have less ICD (i.e. 90% of the time, it is less than 1000 meter) and vice versa. A few notable exceptions are also observed in Figure 8, i.e. some pairs of different places have very less ICD (potentially due to physical proximity of the two places) and some pairs of of same places (nearly 10%) have ICD higher than 1000 meter (potentially due to error from GeoLocation APIs as it is populated using crowd-sourced data). In our experiments, we use 1000 meter as ICD to determine if the two Cell ID clusters belong to the same place.

B. Encounter Prediction Engine

Several studies have shown that given a week worth of mobility profile, it is possible to predict places that are likely to be visited by a person in the future [1]. However, they do not focus on predicting the actual arrival time and time spent at the visited places, both of which are critical for accurately predicting future encounters. Burbey et al [19] used WiFi dataset to show that future places can be predicted with approx. 90% accuracy using a Markov prediction model. However, frequent changes in arrival patterns of a user and predictor based on Markov model did not provide good prediction accuracy for arrival time. Accurate prediction of encounter time is further complicated by the fact that even if one of the users deviates from usual mobility (places or arrival time), it will result in wrong encounter prediction.

Burbey et al [19] proposed "traversing a sequence model" to represent location and time and use it to predict arrival time of users at different locations visited by them. We extend their model to predict encounters between a pair of users, as shown in Figure 9. Prediction of encounter times are done using at least a week worth of past mobility profiles. In the first step, we arrange the mobility profiles into mobility sequences for every day. A mobility sequence contains a series of places visited by a user together with other information such as start time and stay time (duration) at that place. Apart from place specific information, mobility sequences also embed day specific contextual information such as weekday or weekend as a user's mobility may be different on a weekday and a weekend [1], [11].

We define two metrics $support$ and $confidence$ for our encounter prediction model. We use $support$ to find frequent

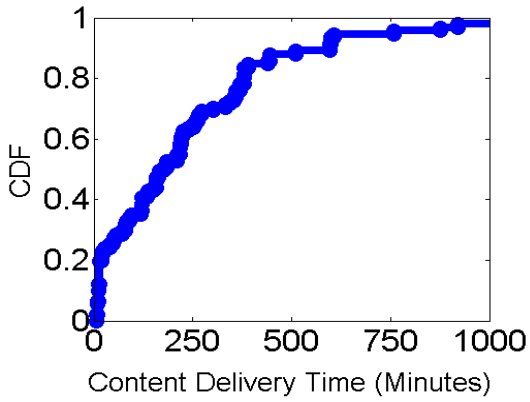


Figure 10: CDF of content transfer delay for all content requests for which encounter was possible within the deadline - nearly 88% requests results in content exchange in less than 10 hours

(regular) places in a user's mobility profile (in other words, eliminate all the places in a user's profile that occur rarely). Similarly, $confidence$ is used to find the most frequent arrival time of a user at a given place.

Following is the step by step explanation of our encounter prediction model for two users A and B:

- $Step1$: From the historical mobility profiles, generate training mobility sequences i.e S_A and S_B given some context c. $S_A = S_{\text{A1}}, S_{\text{A2}}.....S_{\text{Aj}}$ and $S_B = S_{\text{B1}}, S_{\text{B2}}.....S_{\text{Bk}}$.
- $Step2$: Find the set of frequently visited places from i.e FP_A and FP_B respectively for S_A and S_B. For a user, any place P_i is categorized as frequent visited, when $support(P_i) \geq \alpha$. As an example, for user A, $support(P_i) = \frac{count(\forall S_{\text{Ai}} \in S_A where P_i \in S_{\text{Ai}})}{count(S_A)}$. We then generate a new mobility sequence from FP_A and FP_B respectively. Using S_A and S_B, we generate new mobility sequences, S'_A and S'_B, that only contains information pertaining to frequently visited places, defined in their corresponding sets FP_A and FP_B.
- $Step3$: Using FP_A and FP_B, we find a set of encounter places, OP_{AB}, for users A and B.
- $Step4$: For every place in OP_{AB}, we find most frequent arrival time contained in S'_A and S'_B separately using the confidence metric, calculated as $confidence(P_i, T_i) = \frac{count(<P_i, T_i> \in S'_{\text{Aj}} where S'_{\text{Aj}} \in S'_A)}{count(S'_A)}$, where one of the arrival times for user A at P_i is T_i. Using S'_A and S'_B, we then generate new mobility sequences, S''_A and S''_B, that contain information about places from OP_{AB} that have confidence value for their arrival time greater than β.
- $Step5$: Using S''_A and S''_B, encounters at all the overlapping places, E_{AB}, are calculated.
- $Step6$: Given query time, find the next time in E_{AB} to predict encounter time between A and B.

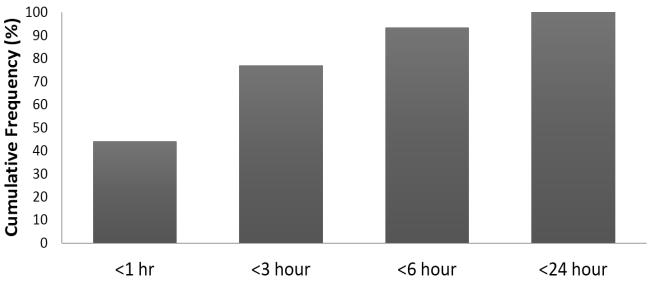


Figure 11: Cumulative frequency (%) of encounter prediction time error. In nearly 77% of the instances where encounter was possible within a day, our model predicted within an error of three hours

C. Evaluation

We evaluate our encounter prediction model using the real content requests, which were made by *MobiShare* participants. Each search entry contains content id, content requester ID (say A), content source ID (say B), and timestamp. Timestamp of search is used as a query time to predict encounters between content requester and provider. For evaluation, we take mobility profile generated using WTCA algorithm. However, same encounter prediction model can use mobility profiles generated using GCA or only WiFi too. Given at least a week of historical mobility profiles for A and B and query time, we predict their next encounter (say T_{ab}), within a day. For accuracy comparison, we found actual encounter time derived from the current mobility profile for that day. In some cases, whenever there is no predicted encounter possible between two users within a day, the encounter time prediction algorithm returns null (i.e. refusing to predict). Empirically, we have found value of α equal to 0.3 and value of β equal to 0.5. α prunes very rarely occurring places from user's mobility profile, for instance given a week's mobility profile history, $\alpha = 0.3$ implies that it will prune any place that occurred less than two times in a week.

From a total of 250 content requests, there were 11 instances (4.4%) where our encounter prediction algorithm predicted a possible encounter with in a day but no actual encounter occurred. For the rest of instances, our model either correctly predicted the encounter or that an encounter will not occur. In a total 182 (72.8% of total requests) instances, where an encounter is predicted, we compare the accuracy of predicted encounter time (T_{ab}) with actual encounter time (T'_{ab}) derived from the mobility profile for that day. Corresponding prediction error is shown in Figure 11. Further, Figure 10 shows that about 88% of the content requests, for which an encounter was possible within a day, resulted into content transfer in less than 10 hours.

V. Related Work

The two closely related areas of work for the *MobiShare* system are (a) Mobility Profiling and Prediction and (b) Opportunistic Content Search and Sharing.

Mobility Profiling and Prediction: With over $100,000$ user mobility traces, Gonzalez et al [1] showed high degree of spatial and temporal regularity in people's movements across days. Similarly, Vu et al [11] used WiFi and Bluetooth traces to characterize movements of mobile users and built a model to predict location, stay duration, and contact with a reasonable accuracy. In *MobiShare*, we use the algorithm proposed by [11] to cluster WiFi APs according to places. Talipv et al [12] collected rich location data using GPS, WiFi, accelerometer, and compass to build a context provider which is able to locate users to room level accuracy and then use this information to predict user's mobility for sharing content opportunistically. Most of feature phone do not have access to rich sensors such as accelerator, compass and GPS. *MobiShare* compromises slightly on accuracy and mostly uses Cell ID and WiFi data to build mobility profile which is energy-efficient and widely available on different phones. Demirbas et al [5] uses Cell ID data to generate spatio-temporal mobility profile of mobile users using MIT's reality mining dataset. However, they mostly take help of manually-tagged places for clustering Cell IDs without any evaluation of the produced clusters where as *MobiShare* clusters Cell IDs without requiring manually tagging and does a comprehensive evaluation using ground truth generated from WiFi data.

Content Search and Sharing: Majority of the content search protocols for mobile networks are distributed. A message containing a search query by the content requester is routed in the network to find the peer with requested content [12]. Content search query and actual content is routed in the network using underlying routing protocols such as Prophet [6], EBR [7], FER [8] and $3R$ [10]. [9] described and compared techniques for limiting content search query in the network, e.g. limited hop count, TTL and first matching drop. McNamara et al [14] described a specific application of media sharing in urban transport such as city subway since users often travel with each other in subway. Many of these proposed systems are only evaluated using simulated data. Basic architecture of *MobiShare* is different from these work and it is implemented and evaluated as an end to end system. Further, *MobiShare* builds upon the observation of McNamara et al that pair of users are often co-located with each other which can be potentially be used for content sharing.

VI. Discussion and Future Work

Previous studies specifically for India have shown that even people with limited academic background transfer multimedia content over Bluetooth [20]. Lack of scalability in the current local sharing mechanisms, for large number of users and diverse content types, is addressed in this work through the proposed *MobiShare* system with the end goal of making content sharing ubiquitous. Unlike previous work in

opportunistic networks,*MobiShare* uses hybrid architecture, where the cloud acts as a control information gateway and facilities sharing between mobile peers. Use of the cloud enables scalable content search and allows content-requester to know in real-time if she can get the desired content and in how much time. Due to limited battery of mobile devices and lack of incentives for relay peers, *MobiShare* currently only allows direct (1-hop) transfer of content between source and destination.

MobiShare uses multiple location interfaces available on most of the phones, e.g. Cell ID, WiFi to build mobility profiles of the users. Our Graph-based Clustering Algorithm (GCA) infers places (Cell ID clusters) using daily cell records and accordingly builds users mobility profiles. Experimental evaluation, using data collected from actual users of *MobiShare* system, shows that GCA produces an average accuracy of 80%. We further refine clusters obtained using GCA with the help of limited WiFi training data to achieve an average accuracy of 87% . Our evaluation on pair-wise content requests in *MobiShare* system showed that approx. 96% of the time, the algorithm was able to accurately predict an encounter in the given deadline and the exact encounter time was predicted with a small error of less than 3 hours in 78% of instances.

For every content request, mobile application in *MobiShare* coordinates with the cloud to identify future encounters and accordingly switches appropriate interface (Bluetooth/WiFi) to exchange the desired content, thus bringing energy efficiency. Use of cloud in *MobiShare* , brings much more control, reliability, and resource-efficiency as compared to distributed architecture. We are working towards extending the basic building blocks of *MobiShare* system discussed in this paper as follows:

1) Extend the encounter prediction model to incorporate, in real-time, deviation in user's mobility using the location updates and accordingly revise the expected encounter time.
2) Allow for multi-hop transfers in *MobiShare*, wherein content requestor can specify constraints such as minimum delivery time, forwarding through a peer who is part of her social network, among others.
3) Incorporate privacy-preserving matching service such as Koi [21] in the cloud which can give same level of privacy guarantees as given by total distributed architectures.

REFERENCES

[1] Barabi A., Understanding individual human mobility patterns," Nature 453, 779-782.

[2] Scott J., Hui P., Crowcroft J., and Diot C.,Haggle: A Networking Architecture Designed Around Mobile Users, IFIP WONS 2006.

[3] Yadav K., Naik V., Singh A., Singh P.,Kumaraguru P., and Chandra U.,Challenges and novelties while using mobile phones as ICT devices for Indian masses: short paper, NSDR'10.

[4] Hui P., Chaintreau A., Gass R.,Scott J.,Crowcroft J., and Diot C., Pocket Switched Networking: Challenges, Feasibility, and Implementation Issues, WAC 2005.

[5] Bayir M.A., Demirbas M.,PRO, and Eagle N.: Discovering spatiotemporal mobility profiles of cellphone users, WOWMOM 2009.

[6] Lindgren A., Doria A., and Scheln O., Probabilistic routing in intermittently connected networks, Mobihoc 2003.

[7] Nelson S. C.,Bakht M., and Kravets R., Encounter-based routing in DTNs, Infocom 2009.

[8] Yuan Q., Cardei I., and Wu J., Predict and relay: An efficient routing in disruption-tolerant networks, Mobihoc 2009.

[9] Pitkanen M., Karkkainen T., Greifenberg J., and Ott J., Searching for Content in Mobile DTNs, PERCOM 2009.

[10] Vu L., Do Q., and Nahrstedt K., 3R: Fine-grained encounter-based routing in delay tolerant networks, WoWMoM 2011.

[11] Vu L., Do Q., and Nahrstedt K., Jyotish: Constructive approach for context predictions of people movement from joint Wifi/Bluetooth trace, PerCom 2011.

[12] Talipov E., Chon Y., Cha H., Content Sharing Over Smartphone-based Delay-tolerant Networks, IEEE TMC,2012.

[13] Friedman R., Kogan A., Krivolapov Y., On Power and Throughput Tradeoffs of WiFi and Bluetooth in Smartphones, INFOCOM 2011.

[14] McNamara L., Mascolo C., and Capra L., Media sharing based on colocation prediction in urban transport, MobiCom'08.

[15] Stuedi P., Mohomed I., Balakrishnan M., Ramasubramanian V.,Terry D.,Wobber T., and Mao Z.M., Contrail: Enabling Decentralized Social Networks on Smartphones, MSR-TR-2010-132.

[16] Carroll A. and Heiser G., An analysis of power consumption in a smartphone. USENIXATC'10.

[17] Gaonkar S., Li J., Choudhury R.R., Cox L., and Schmidt Al, Micro-Blog: sharing and querying content through mobile phones and social participation,MobiSys '08.

[18] Ficek M., Kencl L., Spatial extension of the Reality Mining Dataset. MASS 2010.

[19] Burbey I.E., Predicting Future Locations and Arrival Times of Individuals. Doctoral Thesis, Blacksburg, Virginia, April 2011.

[20] Smyth T.N., Kumar S., Medhi I., Toyama K., Where there's a will there's a way: mobile media sharing in urban india, CHI 2010.

[21] Guha S.,Jain M. and Padmanabhan V.,Koi: A Location-Privacy Platform for Smartphone Apps, NSDI 2012.