Mobility Profiler: A Framework for Discovering Mobile User Profiles (TECHNICAL REPORT Version)

Murat Ali Bayir [a], Murat Demirbas [a], Nathan Eagle [b],

[a] *Department of Computer Science and Engineering, University at Buffalo, SUNY, 14260, Buffalo, NY, USA*

[b] *MIT Media Laboratory, Massachusetts Institute of Technology, 02139, Cambridge, MA, USA*

Abstract

Mobility path information of cellphone users play a crucial role in a wide range of cellphone applications, including context-based search and advertising, early warning systems, city-wide sensing applications such as air pollution exposure estimation and traffic planning. However, there is a disconnect between the low level location data logs available from the cellphones and the high level mobility path information required to support these cellphone applications. In this paper, we present formal definitions to capture the cellphone users' mobility patterns and profiles, and provide a complete framework, **Mobility Profiler**, for discovering mobile user profiles starting from cell based location log data. We use real-world cellphone log data (of over 350K hours of coverage) to demonstrate our framework and perform experiments for discovering frequent mobility patterns and profiles. Our analysis of mobility profiles of cellphone users expose a significant *long tail* in a user's location-time distribution: A total of 15% of a user's time is spent on average in locations that each appear with less than 1% of time.

Key words: Human Mobility, Mobility Mining, city wide sensing, cell phone user profiling

Email addresses: mbayir@cse.buffalo.edu (Murat Ali Bayir), demirbas@cse.buffalo.edu (Murat Demirbas), nathan@mit.edu (Nathan Eagle).

1 Introduction

Cellphones have been adopted faster than any other technology in human history [14], and as of 2008, the number of cellphone subscribers exceeds 2.5 billion, which is twice as many as the number of PC users worldwide [1]. To capture a slice of this lucrative market, Nokia, Google, Microsoft, and Apple have introduced cellphone operating systems (Symbian, Android, Windows Mobile, OS X) and open APIs for enabling application development on the cellphones. Recently, cellphones have also attracted the attention of the networking and ubiquitous computing research community due to their potential as sensor nodes for city-wide sensing applications [18,17,12,39,28,24,29].

Mobility path information of cellphone users play a central role in a wide range of cellphone applications, such as context-based search and advertising, early warning systems [35,5], traffic planning [23], route prediction [30,31], and air pollution exposure estimation [13]. Cellphones can log location information using GPS, service-provider assisted faux GPS or simply by recording the connected cellular tower information. However, since all these location logs are low level data units, it is difficult for the cellphone applications to access meaningful information about the mobility patterns of the users directly. To make mobility data more readily accessible to cellphone applications, higher level data abstractions are needed.

To address this problem, we focus on the problem of discovering spatiotemporal mobility patterns and mobility profiles from cellphone-based location logs. In particular, the contributions of this paper are as follows:

(1) In order to capture the mobility behaviors of cellphone users at a level of abstraction suitable for reasoning and analysis, we introduce formal definitions for the concepts of *mobility path* (denoting a user's travel from one end-location to another), *mobility pattern* (denoting a popular travel for the user supported by her mobility paths), and *mobility profile* (providing a synopsis of a user's mobility behavior by integrating the frequent mobility patterns, contextual data, and time distribution data for the user). Although human mobility has been studied in different contexts in previous work [25,21,34,26], this paper focuses on robust and consistent characterization of mobility behaviors of cellphone users to be employed in very large-scale (city wide) sensing, social networking, and commercial applications.

(2) We design and implement a complete framework, the **Mobility Profiler**, for discovering mobility profiles from raw celltower connection data. Our framework addresses a commonly encountered phenomenon

[1] www.wirelessintelligence.com

in real-world cellular networks, *celltower oscillation*, where even when the user is static she may be assigned to a number of neigboring celltowers for load-balancing purposes or due to changes in the ambient RF environment. Our framework removes oscillation side-effects by determining oscillating celltower pairs from the cellphone logs and grouping them in a single cluster. Furthermore our framework exploits the geometric nature of the problem to improve the performance of the discovery process: our framework first constructs a celltower topology from the available mobility paths and then uses this topology to expedite the pattern discovery process by eliminating a majority of candidate path sequences as unrealizable (due to the topological constraints). In the same vein, our framework introduces new support criterias based on string matching to increase the algorithm's performance during support checks for the mobility patterns.

(3) We validate and demonstrate our framework by using the "Reality Mining" data set [2] containing 350K hours of celltower connection data. Using this dataset, we perform comprehensive experiments to determine the thresholds for when to consider a location as an end-location versus an interim-location on a mobility path. We identify two types of end-locations, observable and hidden, and show that both of them are necessary for correct construction of mobility paths.

(4) Finally, our analysis of the cellphone users' mobility behaviors yields important lessons for networking researchers interested in testing large-scale ad-hoc routing protocols. As also identified in a recent study [21], we find that users spend approximately 85% of their time in 3 to 5 favorite locations, e.g., home, work, shopping. However, our analysis has exposed a more interesting phenomena for the distribution of the remaining 15% of the users' time. We identify a significant *long tail* in a user's location-time distribution: **Approximately a total of 15% of a user's time is spent in locations that each appear with less than 1% of time.** One implication of this finding is that, while simulating/testing large-scale mobile ad-hoc protocols, it is not sufficient to simply take the top-k popular locations. Doing so will discard about 15% of a user's visited locations. We illustrate the importance of this effect in the context of the air pollution exposure estimation application described in section 4.5.

Last but not least, the mobility profiles we generate for cellphone users include temporal information for patterns (which days of the week and which hours of the day) and time distribution data for all locations. These mobility profiles are useful for early warning systems and route prediction applications. By coupling the time-context with the mobility paths, these mobility profiles may be useful for the purposes of synthetic mobility scenario generation research.

[2] http://reality.media.mit.edu

Outline of the paper. The next section explains Reality Mining data set and mobility profiler architecture. Section 3 defines the mobility path concept, gives mobility path construction, mobility pattern discovery method, and construction of mobility profiles. The experimental results on the data set are presented in section 4. Related work is given in section 5, and conclusions in section 6.

2 Preliminaries

2.1 Reality Mining Data Set

The dataset for our work is collected by the Reality Mining project group from MIT Media Labs, that performed an experimental study involving 100 people for the duration of 9 months. Each person is given a Nokia 6600 cellphone with a software that continuously logs data about the location of the cellphone. Due to the lack of GPS in the Nokia 6600, the location is recorded NOT in terms of an exact longitude-latitude pair, but rather in terms of the celltower currently connected. In order to render the celltower ids meaningful, the cellphone software prompts the user to provide a tag when it encounters a new celltower. This way, some celltower locations were able to be tagged semantically with a specific meaning for that user.

The logged data from all the cellphones total around 350K hours of monitoring time and fit into a database of 1GB size. The necessary data for our mobility profiler framework are stored in four tables. Figure 1 shows the database schema that presents the relation between these tables. The Cellspan table stores the connectivity information of a person to a celltower. The Cellname table stores user-specific semantic tags for celltowers. Celltower and Person tables store all the celltower and cellphone user information. The name field in the Celltower table denotes the celltower's broadcasted real name (a numerical id).

2.2 Overview of the Mobility Profiler Framework

Figure 2 illustrates the general architecture of our framework. We start with the "path construction" to construct ordered set of celltower ids that correspond to a user's travel from one end-location to another. Then, we apply "cell clustering" to gather the oscillating celltowers in the same group and replace the celltowers with their corresponding clusters so as to remove the oscillation problems on the paths. After the cell clustering, we apply

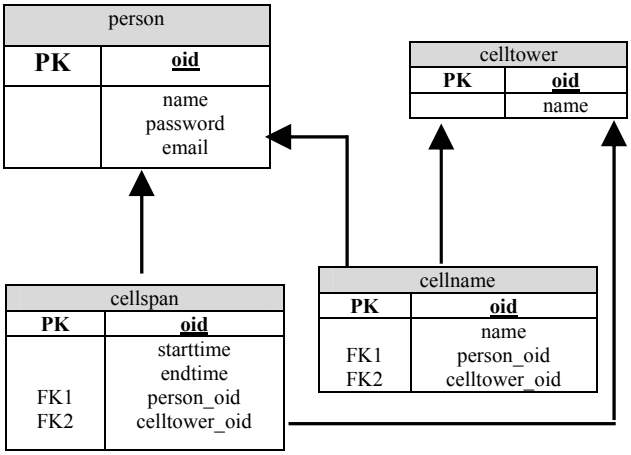

Fig. 1. Mobility Profiler Database

the "topology construction" using the paths of cell clusters as input. The resultant topology information of clusters are employed for eliminating the majority of the candidate path sequences to expedite the "pattern discovery".

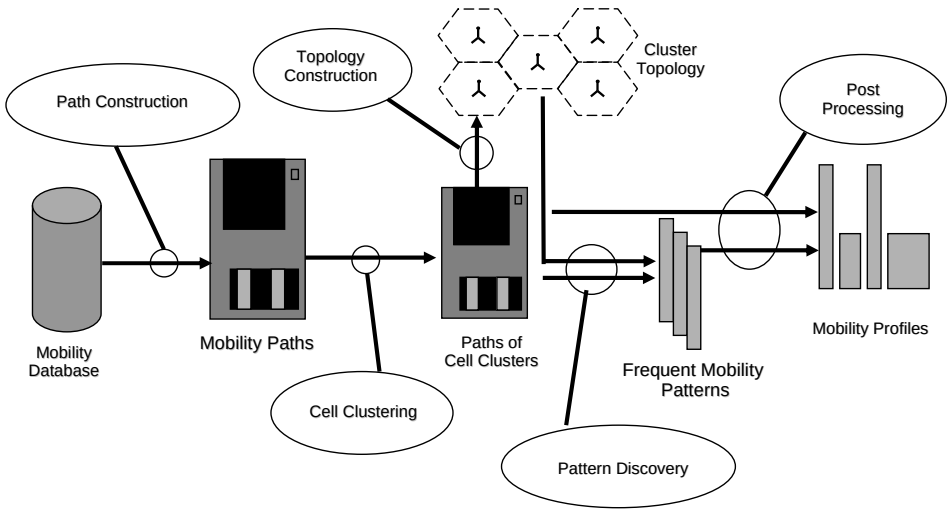

Fig. 2. Mobility Profiler Framework

In the pattern discovery phase, we discover the frequent mobility patterns of each user separately. This task is executed efficiently by employing the topology information and a string matching support criteria (which we discuss later). In the "post processing" phase, we generate cellphone user profiles from their personal mobility patterns by adding the time-context information to the patterns and we generate time distribution data by using paths of cell clusters.

3 Mobility Profiler

In this section we present the five phases of the Mobility Profiler framework in detail.

3.1 Path Construction Phase

Before we proceed to present the construction of the mobility paths for users, we give some basic definitions.

The connectivity information (of a person to a celltower) stored in the Cellspan table is gathered as follows. When a celltower switching occurs, the end time for the previous celltower is captured and a new record is created in the cellphone that contains the start and end time for that previous celltower. Simultaneously, the start time for the new celltower is recorded and is kept until the next celltower switching occurs. There may also be an unaccounted time-gap between two celltower switchings due to disconnection from all base stations or turning off the cellphone. To account for these, we define two time intervals:

Definition (Cell Duration Time): Cell duration time is the difference between end and start time *for each cell span record L*, that represents the connectivity information to a particular celltower. The cell duration time for each cell span record is calculated as:

$$L^k_{dur} = L^k_{end} - L^k_{start} \tag{1}$$

Here L^k_{dur} is the cell duration time for k^{th} cell span record, L^k_{end} is the connection end time and L^k_{start} time is the connection start time for that entry.

Definition (Cell Transition Time): Cell transition time is the difference between the end and start time of *two contiguous cell span record belonging to the same subject in the Reality Mining study (i-th user)*. The cell transition time is calculated as:

$$L^k_{(i)tra} = L^{k+1}_{(i)start} - L^k_{(i)end} \tag{2}$$

Here L^k_{tra} is the k^{th} cell transition time of the user, L^k_{end} is the connection end time for the $(k)^{th}$ cell-span record for that user and L^{k+1}_{start} time is the connection start time for $(k+1)^{th}$ cell-span record for the same user.

Definition (Observed End-Location): An observed end-location record corresponds to a celltower location C_k in the k^{th} cell-span record the duration time of which is greater than a predefined upper bound $\delta_{duration}$:

$$L^k_{dur} > \delta_{duration} \tag{3}$$

To illustrate consider a user arriving to her work place where she stays connected to a celltower for 5 hours. When the user later leaves for home, a cell switching occurs. Since $L_{dur} = 5$ hours is larger than $\delta_{duration}$ time (of say 10 minutes) the cell location C_k is accepted as an end-location and the id of the corresponding celltower is marked as an observed end location.

Definition (Hidden End-Location): A hidden end-location between two contagious cell span record k^{th} and $(k+1)^{th}$ corresponds to a location H_k in which the user stayed longer than a predefined upper bound $\delta_{transition}$:

$$L^k_{(i)tra} > \delta_{transition} \tag{4}$$

This inequality states that a hidden location occurs when a significant amount of time is elapsed during cell transition. To illustrate, consider a user that switches her cellphone at a movie theater and then switches it back on at home after 3 hours. Since the transition time (3 hours) exceeds the threshold $\delta_{transition}$ (say 10mins), we say that the user has been in an unknown hidden end-location H_k for these time intervals. The same case occurs when user is out of cellphone connectivity range for a significant amount of time.

Note that the Cellspan table does not store "related" cell-span records together. The main idea of the mobility path is to group cell span records together to correspond to users' travel from one end-location to another. We define mobility path formally as follows:

Definition (Mobility path): A mobility path $C = [C_1, C_2, C_3, \ldots, C_n]$ is an ordered sequence of celltower ids corresponding to the cells that a user visited during her travel from one end-location to another. The mobility path must satisfy the following two rules:

End Location Rule:

- $\forall C_k \in C, L^k_{dur} > \delta_{duration} \Rightarrow k = 1 \;\; or \;\; k = |C|$

Transition Time Rule:

Table 1
An example cell span data set

oid	p_oid	T_{start}	T_{end}	T_{dur}	T_{tra}	$cell\_id$
1	1	0	4	4	−1	C_1
2	1	6	9	3	2	C_2
3	1	9	13	4	0	C_3
4	1	15	22	7	2	C_5
5	1	23	27	4	1	C_3
6	1	27	30	3	0	C_1
7	1	43	47	4	13	C_2
8	1	49	52	3	2	C_3
9	1	56	58	2	4	C_1
10	1	58	61	3	0	C_3
11	1	62	66	4	1	C_4

- $\forall C_k, C_{k+1} \in C \Rightarrow L_{start}^{k+1} - L_{end}^{k} < \delta_{transition}$

The first rule states that the observed end-locations can only be the first or last locations of the mobility path. Since the paths can also be terminated due to a hidden end-location, the dual of this rule is not true. This rule also implies that for any location that is neither the first nor the last location, the duration time should be smaller than or equal to the predefined maximum cell duration threshold $\delta_{duration}$. The intuition behind this rule is that if a cellphone user stays for a significant amount of time in a cell area C_k, then C_k should be taken as an end-location and the current path should be terminated.

The second rule states that the elapsed time for each celltower transition within the path should not be greater than a predefined threshold $\delta_{transition}$. Thus, a cellphone user can not visit a hidden end-location within the path, otherwise the current path is terminated. The intuition behind the second rule is that if a user stays a significant amount of time outside cellphone connectivity, she may travel to locations that are not captured. In that case, merging hidden locations with previous locations increases the error and leads to noisy data in the paths.

One may argue that there is no need to use transition time threshold and hidden end location concept, instead duration threshold between the starting times of contagious cell span records is sufficient to detect end locations. However, there will be boundary cases in which the sum of contagious du-

Algorithm 1 Mobility Path Construction

1: **Input:** (L, $\delta_{duration}$, $\delta_{transition}$)
2: L: // The set of input records sorted with respect to time
3: δ_{dur}: // upper bound for maximum cell duration time
4: δ_{tra}: // upper bound for maximum cell transition time
5: **global variables:** fSet, tSet // final and temp Path Set
6: **procedure** CreateNewPath (p_oid, cell, start, end)
7: $cellSeq := (cell, start, end)$
8: $tSet := tSet \cup (p\_oid, cellSeq)$
9: **end procedure**
10: **procedure** PathConstruction (L, δ_{dur}, δ_{tra})
11: $fSet := \{\}$
12: $tSet := \{\}$
13: **for each** L_i of L
14: $dur_i := end_i - start_i$
15: **If** $dur_i \leq \delta_{dur}$ **then**
16: **If** $\exists path_k \in tSet$ **and** $p\_oid_k = p\_oid_i$ **then**
17: **If** $(start_i - endTime(path_k)) \leq \delta_{tra}$ **then**
18: $path_k := (p\_oid_k, cellSeq_k \cup (C_i, start_i, end_i))$
19: **Else**
20: $fSet := fSet \cup path_k$
21: $tSet := tSet - path_k$
22: $CreateNewPath(p\_oid_i, C_i, start_i, end_i)$
23: **End If**
24: **Else**
25: $CreateNewPath(p\_oid_i, C_i, start_i, end_i)$
26: **End If**
27: **Else**
28: **If** $\exists\, path_k \in fSet$ **and** $p\_oid_k = p\_oid_i$ **then**
29: **If** $(start_i - endTime(path_k)) \leq \delta_{tra}$ **then**
30: $path_k := (p\_oid_k, cellSeq_k \cup (C_i, start_i, end_i))$
31: $fSet := fSet \cup path_k$
32: $tSet := tSet - path_k$
33: $CreateNewPath(p\_oid_i, C_i, start_i, end_i)$
34: **Else**
35: $fSet := fSet \cup path_k$
36: $tSet := tSet - path_k$
37: $CreateNewPath(p\_oid_i, C_i, start_i, end_i)$
38: **End If**
39: **Else**
40: $CreateNewPath(p\_oid_i, C_i, start_i, end_i)$
41: **End If**
42: **End If**
43: **end for each**
44: **end procedure**

ration and transition time exceed end time threshold, although none of them can not exceed threshold alone. To illustrate; let the time information of two contagious cell span record belonging to same user is given in Table 1.

Assume that $\delta_{duration} = 7$ is used. if we define the cell duration time as the time difference between starting times of contagious cell span records, since $L^{k+1}_{start} - L^{k}_{start} > \delta_{duration}$, current is path is ended after L_k. However, if we use both of the time constraints and take $\delta_{duration} = \delta_{transition} = 7$, we do not need to end current path after L_k since the following conditions are satisfied:

- $L^{k}_{end} - L^{k}_{start} > \delta_{duration}$
- $L^{k+1}_{start} - L^{k}_{end} > \delta_{transition}$
- $L^{k+1}_{end} - L^{k+1}_{start} > \delta_{duration}$

Algorithm 1 presents our path construction. To illustrate, we provide an example execution of the algorithm on the cell-span records given in Table 1. T_{start} and T_{end} correspond to start and end of connection times to the corresponding celltower in each cell-span record. $T_{duration}$ and $T_{transition}$ times are calculated according to the definitions of cell duration and cell transition times. The transition time of the first record is -1 since we do not have any cellspan record before that record. Let $\delta_{duration} = 7$ and $\delta_{transition} = 5$.

After processing the first record, the algorithm creates an initial path containing only the first celltower, $[C_1]$. The algorithm terminates the current path with the cellspan record $oid = 4$, since there $T_{duration} > \delta_{duration}$. Thus, the current path $[C_1, C_2, C_3, C_5]$ is written to the database.

Since the end-location $[C_5]$ is an observed end-location, the new path is initialized as $[C_5]$. The algorithm continues until cellspan record $oid = 7$, where $T_{transition} > \delta_{transition}$. The algorithm terminates the current path $[C_5, C_3, C_1]$ before appending the current celltower C_2. Since the user enters a hidden location after cell C_1, C_2 is not appended to the previous path and a new path $[C_2]$ is initialized. The algorithm then continues to process cell-span records until all records are exhausted. When the algorithm stops, the the mobility paths in Table 2 are generated:

Table 2
Reconstructed Paths Database

PathId	*Path*
1	$[C_1, C_2, C_3, C_5]$
2	$[C_5, C_3, C_1]$
3	$[C_2, C_3]$
4	$[C_1, C_3, C_4]$

3.2 Cell Clustering

A major problem with the cellular network connectivity data is that a cellphone may dither between multiple cells even when the user is not mobile. A similar problem is also addressed in the Wi-Fi networks referred as the ping-pong effect [32] which is attemped to remove by detecting two types of oscillating patterns by considering general geometry of cell range without using real locations.

Since we have the location information of cell towers partially, we have a two phased approach to solve this problem. In the first phase, we have clustered the cell towers which has already location tags generated by users. Each cluster is formed with respect to location information of celltowers on the map. In the second level, we handle the the remaining untagged celltowers by identifying oscillating celltower pairs. After that, each untagged celltower is assigned to a cluster by considering its oscillating pair information.

We define an oscillating cell pair as the ones that have k mutual switches with each other in mobility paths. For example, given a mobility path $P = [x, y, x, y, w, v, w]$ and minimum switching count $k = 3$, $< x, y >$ becomes the only oscillating pair. The first switch occurs from x at $index = 1$ to y at $index = 2$, the second switch from y at $index = 2$ to x at $index = 3$, and finally, the third switch occurs from x at $index = 3$ to y at $index = 4$. Due to the space limitations we relegate the details of our algorithm for identifying the oscillating pairs in a given mobility path to our technical report.

After identifying the oscillating pairs in the mobility paths, we assign untagged cell towers to the current clusters generated by using tagged cell towers. Each new celltower is assigned to cluster which contains the maximum number of oscillating pairs. The idea comes from the fact that each celltower oscillates with the ones that is geographically close to itself. If every cluster has no oscillating pair for the current tower, an untagged new cluster is created with the current celltower only. After assigning all all cell towers to clusters, each cell tower in the mobility paths is replaced by its corresponding cluster. By this way, we obtain mobility paths of clusters instead of cells.

3.3 Topology Construction

Topology construction is used for eliminating majority of candidate path sequences during the pattern discovery phase. In general, pattern discovery problem is solved by an exponential time algorithm, which may take a

significant amount of time to execute. By employing the cell cluster neighborhood topology during pattern discovery, the candidate sequences which can not possibly correspond to a path on the cell cluster topology graph can be eliminated without calculating their supports.

The topology construction method is given in Algorithm 2. Since we have user mobility paths as input, the cell cluster topology construction is an easy process by one scan through these paths. Algorithm 2 creates an edge between the cell cluster pairs C_k and C_{k+1} if both of them exist in any path in contagious positions.

Algorithm 2 Topology Construction

```
1:  Input: S: The Set of all paths in terms of clusters
2:  procedure createTopology (S)
3:    TopologyMatrice[][] := null
4:    For Each S_i  of  S // S is whole set
5:        for each (C_k  and  C_{k+1}) ∈ S_i
6:          If TopologyMatrice[C_k][C_{k+1}] = null then
7:            TopologyMatrice[C_k][C_{k+1}] = true
8:          end If
9:        end For Each
10:   end For Each
11: end procedure
```

3.4 Pattern Discovery

In this phase, frequent mobility patterns are discovered from mobility paths. Although not the most recent or the most efficient one in the literature, we use a modified version of the AprioriAll[2] technique. This technique is suitable for our problem since we can make it very efficient by pruning most of the candidate sequences generated at each iteration step of the algorithm using the topological constraint mentioned above: for every subsequent pair of cell-clusters in a sequence, the former one must be neighbour to the latter one in the cell-cluster topology graph. We call this new version of AprioriAll as Sequential Apriori Algorithm. An important criteria in our domain is that a string matching constraint should be satisfied between two sequences in order to have support relation. For example, the sequence $< 1, 2, 3 >$ does not support $< 1, 3 >$ although 3 comes after 1 in both of them. However, sequence $< 1, 3, 2 >$ supports $< 1, 3 >$. A path S supports a pattern P if and only if P is a subsequence of S not violating the string matching constraint. We call all the paths supporting a pattern as its support set.

Sequential Apriori Algorithm (Algorithm 3): In the beginning, each cell cluster with sufficient support forms a length-1 supported pattern. Then, in

the main step, for each k value greater than 1 and up to the maximum reconstructed path length, candidate patterns with length k+1 are constructed by using the supported patterns (frequency of which is greater than the threshold) with length k and length 1 as follows:

- If the last cell cluster of the length-(k) pattern is incident to the cell cluster of the length-1 pattern, then by appending length-(1) cell cluster, length-(k+1) candidate pattern is generated.
- If the support of the length-(k+1) pattern is greater than the required support, it becomes a supported pattern. In addition, the new length-(k+1) pattern becomes maximal, and the extended length-(k) pattern and the appended length-(1) pattern become non-maximal.
- If the length-(k) pattern obtained from the new length-(k+1) pattern by dropping its first element was marked as maximal in the previous iteration, it also becomes non-maximal.
- At some k value, if no new supported pattern is constructed the iteration halts.

Note that in the sequential Apriori algorithm, the patterns with length-k are joined with the patterns with length-1 by considering the topology rule. This step significantly eliminates many unnecessary candidate patterns before even calculating their supports, and thus increases the performance drastically.

An auxiliary function Support(I:Pattern,S) determines whether a given pattern has sufficient support from the given set of reconstructed user paths. Support of a pattern I is defined as a ratio between the numbers of reconstructed paths supporting the pattern I, the number of all paths.

$$Support(I, S) = \frac{|\{S_i | \forall i \; I \; is \; substring \; of \; S_i\}|}{|S|} \tag{5}$$

In order to make the Sequential Apriori algorithm more understandable, we give an example execution over the constructed paths in the example in Table 2. Let δ=0.25 be taken as minimum support for the Sequential Apriori algorithm. Then, the execution of the sequential apriori technique will generate patterns with their frequencies in four iterations as it is shown in Table 3.

In this table, the patterns in the lower row of each iteration are eliminated due to their insufficient support. The maximal frequent patterns are shown in bold in Table 3. Since at iteration 5, there are no remaining frequent patterns, the algorithm stops.

Algorithm 3 Sequential Apriori

```
1: input: Minimum support frequency: δ, Paths of clusters: S
2: Topology Matrix: Link, The Set of all Cell Clusters: C
3: output: Set of maximal frequent patterns: Max
4: procedure sequentialApriori (δ, S, Link, C)
5:   L_1 := {} // Set of frequent length-1 patterns
6:   for i:=1 to |C| do
7:       L_1 := L_1 U [C_i] | if Support([C_i],S) > δ
8:   for k = 1 to N − 1 do
9:       if L_k = {} then
10:        Halt
11:      else
12:        L_{k+1} := {}
13:       for each I_i ∈ L_k
14:         for each C_j ∈ C
15:           if Link[LastCluster(I_i), C_j] = true
16:             T := I_i • C_j // Append C_j to I_i
17:             if Support(T, S) > δ then
18:                 T.maximal := TRUE
19:                 I_i.maximal := FALSE // since extended
20:                 V := [T_2, T_3,. . . , T_|T|] // drop first element
21:                if V ∈ L_k then
22:                    V.maximal := FALSE
23:                    L_{k+1} := L_{k+1} U {T}
24:                end if
25:             end if
26:           end if
27:         end for each
28:       end for each
29:     end if
30:  end for
31:  Max := {}
32:  for k := 1 to N − 1 do
33:      Max := Max U {S | S ∈ L_k and S.maximal = true }
34:  end for
35: end procedure
```

3.5 Representing Mobility Profiles

Frequent mobility patterns containing only location information and lacking any time-context information are inadequate for several applications, including route prediction, early warning systems, and user clustering. Therefore, we add time-context information to the frequent patterns in order to represent mobile user profiles.

Table 3
Patterns Generated at each Iteration

Step	Patterns	Frequencies
1	$\{< C_1 >, < C_2 >, < C_3 >, < C_4 >, < C_5 >\}$	{0.75, 0.50, 1.00, 0.25, 0.25} ≥ 0.25
2	$\{< C_1, C_2 >, < C_1, C_3 >, < C_2, C_3 >, < C_3, C_1 >, < C_3, C_4 >, < C_3, C_5 >, < C_5, C_3 >\}$	{0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25 0.25} ≥ 0.25
	$\{< C_2, C_1 >, < C_3, C_2 >, < C_4, C_3 >\}$	{0.0, 0.0, 0.0, 0.0} < 0.25
3	$\{< C_1, C_2, C_3 >, <$ **C1, C3, C4** $>, < C_2, C_3, C_5 >, <$ **C5, C3, C1** $>\}$	{0.25, 0.25, 0.25, 0.25} ≥ 0.25
	$\{< C_1, C_3, C_2 >, < C_1, C_3, C_5 >, < C_2, C_3, C_1 >, < C_2, C_3, C_4 >, < C_3, C_1, C_2 >, < C_5, C_3, C_2 >, < C_5, C_3, C_4 >\}$	{0.0, 0.0, 0.0, 0.0, 0.0 0.0, 0.0, 0.0} < 0.25
4	$\{<$ **C1, C2, C3, C5** $>\}$	{0.25} ≥ 0.25
	$\{< C1, C2, C3, C4 >, < C5, C3, C1, C2 >\}$	{0.0, 0.0} < 0.25

Definition (Mobility Profile): A mobility profile for a cellphone user includes personal mobility patterns with contextual time data and distribution of spatiotemporal locations for that user. The time contextual data for mobility patterns are specified in two dimensions:

- **Days of Week:** Each frequent pattern stores its distribution over days of week. That means, the frequent pattern is tagged with the number of its instances observed on each day of the week.
- **Time Slices:** Each frequent pattern stores its distribution over each time slices given in the set {[12:00 a.m., 6:00 a.m.], [6:00 a.m., 12:00 p.m.], [12:00 p.m., 6:00 p.m.], [6:00 p.m., 12:00 a.m.]}. That means, the frequent pattern is tagged with the number of its instances started on each of these time slices.

Apart from the spatiotemporal mobility patterns, mobility profile of each user contains time distribution data of all locations visited by current user. The time distribution data is very important since it identifies the importance

of each location that is proportional to the time spend on them.

4 Experimental Results

In this section, we will present our experimental results on MIT reality mining data set containing 350K hours of cellspan data. For analyzing MIT Reality Mining data, we have implemented Mobility Profiler Framework on Java Environment. The size of the source code for the whole framework is around 4KLOC. Our implementation contains separate module for each of the phases discussed above.

The rest of this section is given as follows: First, we give our results for determining duration and transition threshold, that are used for constructing mobility paths. For cell-clustering, we give our analysis for finding minimum switch count. For the pattern discovery phase, we present examples of interesting patterns discovered from Reality Mining data and give a case study for representing mobile user profile. We have also provide an interesting results related to the average time distribution of the locations for all users. Finally, we present an application of mobility profiles discovered by our framework in the context of air pollution exposure risk estimation.

4.1 Determining End Location Thresholds

As it is mentioned in section 3, path reconstruction process needs three input items which are L, $\delta_{duration}$, $\delta_{transition}$. Therefore, we need to determine $\delta_{duration}$ and $\delta_{transition}$ before executing Path construction process on cell span data L. These two threshold values are determined by analyzing the ratio of cell span record or cell span transitions that is smaller than predefined time values in experiment space. For determining $\delta_{duration}$ time, we have defined our experimental duration time space as a set $\{1, 5, 10, 15, 20, 25, 30\}$ which contains 7 different time values from 1 minute to 30 minutes. After that, we evaluate the ratio of cellspan records the duration time of which is smaller than these 7 discrete values in our experiment set. The result of this first experiment is given in Figure 3. In this graph, the point with the duration threshold $30min$ and $ratio = 0.97$ means the duration time of 97% of all cell span logs is smaller than 30 minutes. As it is easily seen from the graph that the value for all of duration threshold between $[30, infinity)$ lies between $[0.97, 1.00)$. It is obvious that there is no significant difference between any arbitrarily large threshold value >> 30 min (where user is static obviously) and 30 minutes in terms of log ratio. In fact, the line has very small tangent after duration time=10 min which has ratio value of 0.94.

However, if we analyze the left part of duration threshold=10 min. There is significantly sharp switch between two points having duration time=10 min and duration=5 min. In fact, the first sharp decrease occurs when we switch from $10min$ to $5min$. There exists approximately 10% difference between these points. Therefore, we decided to accept the static time threshold as $\delta_{duration}$=10 min.

One can argue that there may be non-static locations in which cellphone user stays more than 10 minutes. To illustrate; a user may wait 15 minute in bus stop which can be intermediate location during trip from school to home. However, as it is shown from our graph, this type of behavior shows rarely since all of the locations the duration time of which is greater than 10 min $[10, infinity)$ lies between [0.94, 1.00) in terms of log ratio. Therefore, we accepted that 10 minutes is a reasonable threshold for $\delta_{duration}$ time. Since our data size is very huge (2.5M of cellspan records), we believe that our graphs gives significant information cellphone users behavior in general.

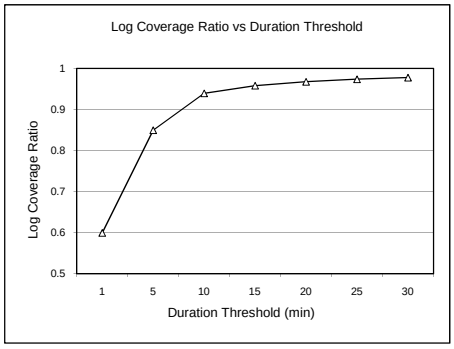

Fig. 3. Duration Time Analysis

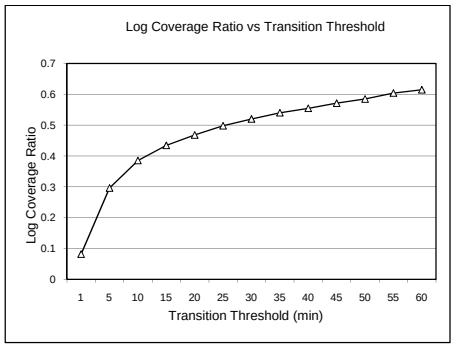

Fig. 4. Transition Time Analysis

For determining $\delta_{transition}$ time, we define our experimental space as a set with 13 different time values from 1 minute to 60 minutes. We do not take higher values than 60 minutes since it is reasonable to accept the existence of hidden end locations if transition time is more than 60 minutes. In order to find acceptable value for $\delta_{transition}$ time, we use the ratio metric that is mentioned above for analyzing $\delta_{duration}$ time. Unlike the analysis of $\delta_{duration}$ time, there is still some visibility problem if we analyze this data without filtering the regular handoffs that take 0 seconds. In reality mining data set, nearly, 99.2% of contiguous cellspan records has regular handoff value that is 0 second that means the cellphone handles 99.2% of celltower switches immediately. It is obvious that the user can not be in hidden end location in this time range. Therefore, we filter regular handoff times for analyzing $\delta_{transition}$. The result of the second experiment is given in Figure 4. In this graph, we notice that the tangent of line after threshold time 10 minutes is greater than one in the Figure 3 for $\delta_{duration}$ time. However, we notice that the tangent of the line is constant after 10 minutes threshold time until 60 minutes. In each neighbor point after 10 minutes, the increase in the log

coverage ratio is around 2-3%. When we analyze the left part of transition threshold=10 min, we see a significantly sharp drop of about 10%. Thus, we accept 10 minutes as a reasonable threshold for $\delta_{transition}$ time. This is also a good choice as it relates to the duration time threshold for determining end-locations.

4.2 Cell Clustering

After determining $\delta_{duration}$ and $\delta_{transition}$ values as 10 minutes, we executed the path construction phase over 2.5M cell-span records resulting in approximately 120K mobility paths. However, these paths included a significant amount of noisy data due to celltower oscillations not correlated with human mobility.

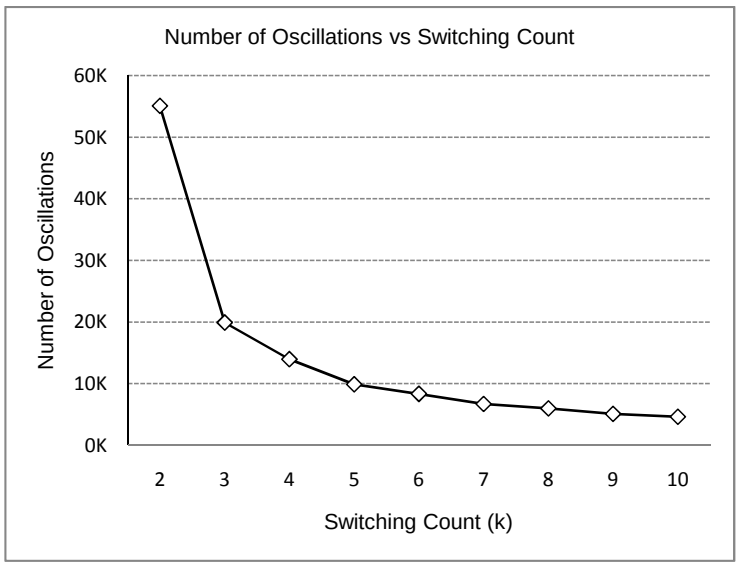

Fig. 5. Switching Count Analysis

For solving the oscillation problem mentioned above, we cluster the celltowers by using their location tags. Each cluster is named by using majority voting over the locations names of its celltowers. For assigning untagged celltowers to the clusters, oscillating pairs of untagged celltowers are discovered. As it is mentioned in clustering section we need minimum swithcing count to find the oscillating pairs. Therefore, we have performed an experiment on determining minimum switching count k. In this experiment, we count the number of oscillations with respect to different switching counts from $k = 2$ to $k = 10$. The results of this experiment is provided in Figure 5. As seen from Figure 5, the tangent of the plot-line decreases as k becomes larger. In fact, when moving on the x axis from infinity to zero. The biggest jump occurs when switching from point $k = 3$ to $k = 2$. We belive that the

number of oscillations due to natural user mobility (which should be distinguished from celltower oscillations) significantly contributes for $k = 2$. Thus, in order to better distinguish between oscillations due to user mobility and celltower oscillations, we take the minimum switching threshold $k = 3$.

After determining oscillating cell cluster by using $k = 3$ as the switching threshold, we find the oscillating pairs of untagged celltowers. Each celltower is assigned to cluster having maximum number of oscillating pairs containing corresponding celltower. If every cluster has no oscillating pair, an untagged new cluster is created with the current celltower only. We found that the average coverage value for the generated clusters is fairly good which is approximately 0.80 and the standard deviation is around 0.08, which means that the majority of coverage values lies in the interval $[0.72, 0.88]$.

4.3 Finding Maximal Mobility Patterns

We executed the pattern discovery phase for generating both global and personal frequent patterns. For the global pattern discovery, we have used frequency support $\delta = 0.001$ which means that each pattern should exist in at least 120 path over 120K total paths to be considered. Since we deal with multiple users for global pattern case, a same celltower with in a cluster can be named differently by each person. In addition, there may be different celltowers having different names in the same cluster. In this case, the name for each cell cluster is determined by using majority voting over celltower names within the cluster.

Table 4
Global Mobility Patterns

Pattern Name	**Frequency**	**Length**
<Home, Media Lab>	0.0267	2
<Media Lab, Home>	0.0267	2
<Home, MIT, Student center>	0.0096	3
<Student Center, MIT, Home>	0.0071	3
<Anils Sofa, Tang>	0.0061	2

An interesting subset of most frequent global patterns are provided in Figure 4. Since the frequency of mobility paths is inversely correlated with the path-length, the size of most frequent paths are usually one or two hops like in the Figure 4. However, the overall distribution of path length is more distributed which is given in Figure 6. As it is easily seen from the figure, more than 80% of the patterns has hop count between 1 and 6. Apart from

pattern length, we have also measured the effect of frequeny threshold on the average size of mobility patterns. Figure 7 shows our results in exponential scale. It is easily seen form the results that, the average size of mobility patterns increases when frequency threshold decreases exponentially. For our global pattern discovery experiment with $\delta = 0.001$, the average pattern size is around 4.8 which means that average hop count for mobility patterns is around 3.8.

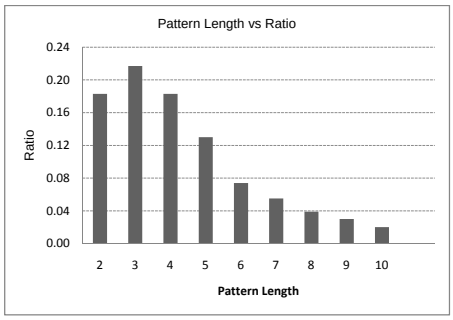

Fig. 6. Pattern Length Analysis

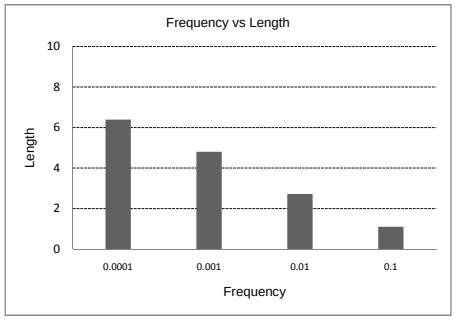

Fig. 7. Frequency vs Len. Analysis

Unlike the global case, personal pattern discovery is more consistent since each celltower is tagged homogeneously by same person. For presenting personal patterns, we choose the paths of single cellphone user as a case study. The number of paths for selected cellphone users is around 2K. Therefore, we choose the frequency threshold as $\delta = 0.005$ which means that each pattern should exist in at least 10 mobility paths. The top 5 five mobility patterns for our case study are given in Figure 5.

4.4 Representing Cellphone User Profiles

Here we present our experimental results for mobility profiling on user X. The top five mobility patterns are plotted in Figure 8 and 9 on two different time domains (day of weeks and time slices). We also analyzed spatiotemporal distribution of visited locations for user X in Figure 10.

Table 5
Top-5 Mobility Patterns of user X

Id	Pattern Name	Frequency
1	<Home, Media Lab>	0.279
2	<Media Lab, Home>	0.265
3	<XXX CommonWealth, Media Lab>	0.133
4	<Home, Charles Hotel, Media Lab>	0.060
5	<Media Lab, Charles Hotel, Home>	0.053

Figure 8 shows the distribution of all five patterns over weekdays and weekends. All of the top-5 patterns are active on weekdays with a balanced distribution over the 5 work days. The peak time for the first, second, and fourth patterns are afternoons whereas the peak time for the third and fifth patterns are evenings (Figure 9).

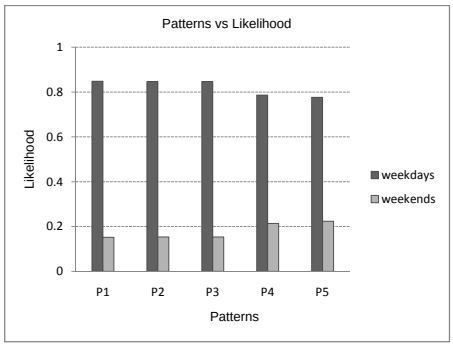

Fig. 8. Days of Week Analysis

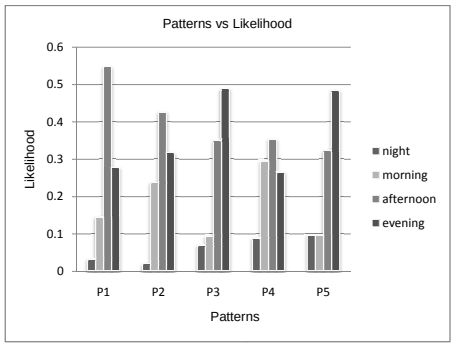

Fig. 9. Time Slice Analysis

As mentioned in section IV, the user profiles give significant information about cellphone user behaviors. For example, on a Tuesday afternoon if user X is at cell area tagged as ”XXX CommonWealth,”, with high probability she will go to cell area tagged ”Media Lab” next. It is very clear that our mobility profiles have potential of producing more correct results for location prediction problem with their additional time dimension.

We have also analyzed the spatiotemporal distribution of locations for user X in Figure 10. Although it may first appear that there is no need to construct mobility paths and perform clustering to extract these spatiotemporal locations, mobility path construction is a very important step for generating an accurate and noise-free time distribution chart, and we have used the mobility paths for user X for constructing the time distribution chart. Mobility paths gather related cell span connectivity records together, and makes it possible to determine and analyze the oscillations and clustering among the celltowers. Replacing cell towers with corresponding clusters within these paths enables us to calculate the time elapsed on each cluster location accurately for the time distribution char.

Figure 10 shows that user X spends 67% of her overall time at home or work. In fact, 79% of overall time elapsed at 8 different locations for user X. An even more interesting phenomenon is found when we consider the distribution of the remaining 6% (others) for user X in Figure 10. These remaining 6% of user X’s time is spent in locations that each appear less than 1% of time: there are 69 different locations for user X in that portion. In other words the spatiotemporal distribution for user X shows a very heavy/long tail. We corroborated this finding in all users’ spatio temporal distributions: **approximately 15% of the users time is spent in a large**

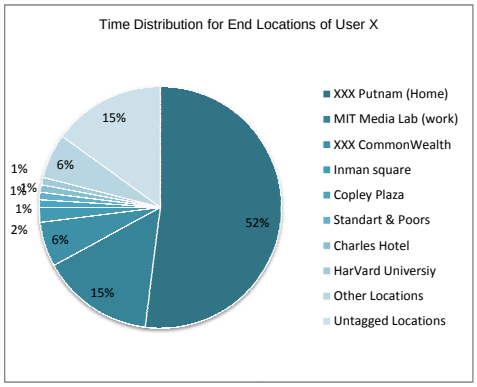

Fig. 10. Time distribution for end locations for user X

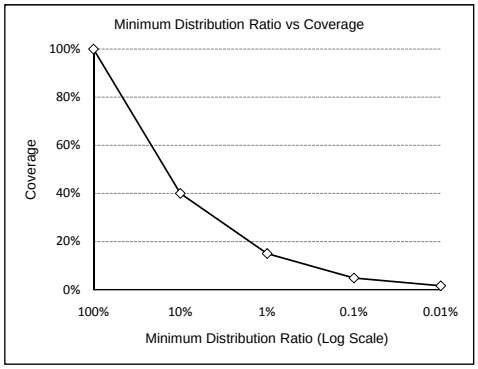

Fig. 11. Minimum Distribution Ratio vs Coverage

variety of locations that each appear less than 1% **of total time**. We present a graph of the number of locations with respect to coverage ratios in Figure 11. In this figure a point (1%, 15%) means that on average 15% of total time elapsed on the locations in which the user spend less than 1% of total time. Since this graph is in logarithmic scale, it is possible to see clearly that there is a 15% heavy tail after 1% minimum distribution ratio. Indeed, the coverage ratio approaches zero only after two more logarithmic scales from that point. The average number of locations that remain in the 15% heavy tail area is more than 800, whereas it is around 12 for the remaining 85% portion.

One implication of this find is that, while simulating/testing large-scale mobile ad-hoc protocols, it is not sufficient to simply take the top-k popular locations. Doing so will discard about 15% of a user's visited locations.

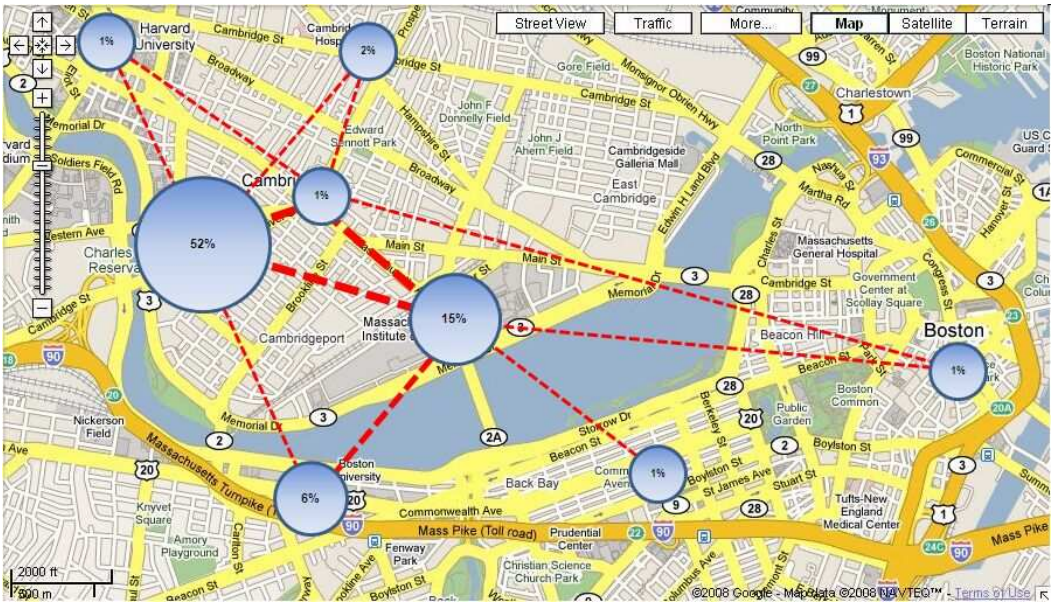

Fig. 12. Time distribution for end locations on map for user X

We are currently using the Reality Mining data for an air pollution exposure estimation application [13]. Estimating air pollutant exposure is not an easy task since air pollution is usually highest in wide urban areas. Many air pollutant concentrations, particularly those related to vehicular traffic, vary as much within cities as they do between cities. The previous modeling approaches for estimating air pollutant exposures of the individual use the residential address [1]. Investigators have attempted to incorporate time-activity data into air pollutant estimation procedures by interviewing study participants regarding their travel schedules [27], filming children to estimate their exposures to indoor sources of pollution (cooking fires)[6], and modeling time-activity patterns in GIS using self-reported travel characteristics [22]. These methods are too costly and time-consuming to apply to large populations. Moreover as we show in Figure 10, since human mobility has a heavy tail, it is infeasible to reach 100% coverage with these approaches, as these approaches capture only the top-k locations, which make up only about 85% of total time.

As an alternative to these methods, we use the spatiotemporal distribution of locations of a person we obtain from the mobility paths. We will integrate these time distribution data with the data obtained from $PM_{2.5}$ air pollution sensors from the Boston area. These sensor data are publicly available at no cost from governmental web sites, such as Department of Environmental Conservation website, U.S. Environmental Protection Agency and U.S. Census Bureau Geography Division website. Since we know the location of each $PM_{2.5}$ sensor, it is feasible to estimate average $PM_{2.5}$ exposures of individuals by calculating weighted average of their spatiotemporal distribution of locations with respect to locations of $PM_{2.5}$ sensors. As an example case study, we graph the location distribution of user X over the Boston area map Figure 12. (For the sake of simplicity the graph shows only the top locations for user X.) The weight of each edge in the graph is proportional to the frequency of the current mobility paths between two locations. The mobility path information allows us to determine the time and routes for when the user is driving/travelling between end-locations. Although the user spends 85% of total time in top locations such as home and work locations, the air pollution exposure risk is higher when she is traveling. This emphasizes the importance of capturing the remainning 15% locations and discovering users' mobility path.

A potential application of our framework is for enriching the content of the social networks web sites, such as facebook and myspace, with the mobility information of users. These social networking sites may present the user with meeting opportunities to other users that have similar mobility profiles to theirs, or suggest places to visit based on the locations recently visited by their mobility-profile-proximity peers.

Another useful application is for estimating better quotes by the car insurance companies. The current cost estimation models for car insurance only takes residential information into consideration. However, cost of the insurance may significantly vary if the users mobility information and time distribution data is known before hand.

Finally, enhancing the performance of peer to peer sharing programs on cellphones with the aid of mobility information is an interesting problem to consider. One can design a peer to peer server which indexes only the names of shared files over users with respect to their location and the mobility information.

5 Related Work

There are several recent works on the benefits of using cellphones as sensor nodes for city-wide sensing applications [18,17,12,39,28]. Researchers also started to investigate models and architecuture for collecting data from privately hold mobile sensors. Karause et al. [29] propose a model for community sensing that enables to share data from personal sensors like cameras or cellphones. They have showed feasibility of their approaches on a traffic monitoring case study. Hull et al. [24] designed CarTel systems that has a GPS sensors and cameras on cars to monitor their movements and send this via opportunistic message forwarding.

In the recent works, cellphone based location data was used for mining human behaviors and social networks analysis [15,40,36]. These works include finding social patterns in user's daily activity, extracting relationship among individuals and identifying socially important locations. Another interesting application of cell based location data is the opportunistic message forwarding [16,11,41,10]. The opportunistic message forwarding is performed by analyzing similarity of individual's mobility behaviors with respect to locations they have visited frequently.

Mobile Landscape project [8] is one of the most comprehensive city wide application in which the celltower location data is analyzed for visualization of population migration and traffic density. Another work similar to ours is carried by Context group from University of Helsinki. They have provided the solution for clustering and route prediction problem for mobile users by using cell based location data [30,31,3]. These works include the definition of user routes from cellular data; however, they do not investigate modeling of mobility.

Human mobility is also used for optimizing load balancing, resource consumption, paging overhead and network planning in cellular networks. MarkouDiakis et al. [34] proposes a hierarchical mobility model for optimizing network planning and handover rate in celluar environments. Their hierarchical model analyzes human mobility in three levels which are City Area, Area level and Street Unit levels. Zanoozi et al. [42] analyzes human mobility inside the single cell for optimizing cell residence time. Liu et al. [33] propose a mobility prediction model for optimizing cell handover residence time. Their method employs Markov Model and Kalman Filter to predict when a mobile node crosses cell boundaries. Bhattacharya et al. [7] utilized prediction model to reduce paging overhead in celluar networks by limiting the number of possible cells that user may enter. Akyildiz et al. [4] proposes a method for predicting future location of mobile node by using moving direction, velocity, current position and historical records. Their results showed that proposed model increase the performance of network in terms of location tracking cost, delays, and call dropping/blocking probabilities. Cayirci et al. [9] showed how mobility pattern of mobile can be used to optimize location update in celluar networks.

Human mobility has been a focus of interest by recent work in wireless networks and ubiqitious computing research community. Musolesi et al. [37] present an extensive survey on mobility models. They divide general mobility models into two categories called traces and synthetic models, the latter being more common due to the difficulty in gathering publicly available traces. Garetto et al. [19], Hsu et al. [26] and Lee et al. [32] propose models for human mobility in Wi-Fi environments. Rhee et al. [25] analyzed human mobility by using GPS data and they proposed that human mobility shows levy walk behaviour. Ghosh et al. [20] examines the human mobility based on semantically related locations forming orbits at different hierarchies by using location data obtained from GPS. Nurmi et al. [38] proposed clustering methods for finding important locations of cell phone users. Their approach uses cell based location data and models the cell tower network as graph based on cell transitions.

In the very recent work, Gonzalez et al. [21] analyzed the mobility patterns of 100K mobile phone users by using cell based location data. Unlike the

Levy walk nature of human mobility [25], that study proves that human trajectories show a high degree of temporal and spatial regularity. They showed that each cell phone user tends to move between most important locations (namely top-k locations). Their findings are also supported by our work since we show that an average 85% of total time are observed in the top locations of the users and the most frequent mobility patterns are the ones between these top locations.

6 Conclusion and Future Work

In this paper, we have proposed a complete framework for discovering mobile user profiles. We have defined the mobility path concept for cellular environments and introduced a novel path construction method. We have also proposed a cell clustering method that provides robustness against noises, such as celltower oscillations and improper handoffs containing time delays. From the experimental results over 350K hours real data, we have shown that our framework is capable of producing user profiles that can be used for city wide sensing applications like air pollutant exposure estimation. Our analysis also discovered a long tail for human mobility behavior: approximately 15% of a person's time is spent in a large variety of locations each of that takes less than 1% time.

As future work, we are going to work on a similar framework that uses GPS data to discover mobile user behaviors. We will also investigate the opportunities for using our mobility profiles in new applications, such as social networking, car insurance estimation and peer to peer file applications over smartphones.

References

[1] S. D. Adar and J. D. Kaufman. Cardiovascular disease and air pollutants: evaluating and improving epidemiological data implicating traffic exposure. *Inhal. Toxicol.*, 19(1):135–149, 2007.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. In *ICDE*, pages 3–14, 1995.

[3] S. Akoush and A. Sameh. Mobile user movement prediction using bayesian learning for neural networks. In *IWCMC*, pages 191–196, 2007.

[4] I. F. Akyildiz and W. Wang. The predictive user mobility profile framework for wireless multimedia networks. *IEEE/ACM Trans. Netw.*, 12(6):1021–1035, 2004.

[5] B. Barnes, A. Mathee, and K. Moiloa. Assessing child timeactivity patterns in relation to indoor cooking fires in developing countries: a methodological comparison. *Int. Journal Hyg. Environ. Health*, 208(3):219–225, 2005.

[6] A. Bhattacharya and S. K. Das. Lezi-update: An information-theoretic framework for personal mobility tracking in pcs networks. *Wireless Networks*, 8(2-3):121–135, 2002.

[7] S. H. C. Ratti, A. Sevtsuk and R. Pailer. *Mobile landscapes: Graz in real time http://senseable.mit.edu/graz/.*

[8] E. Cayirci and I. F. Akyildiz. User mobility pattern scheme for location update and paging in wireless systems. *IEEE Trans. Mob. Comput.*, 1(3):236–247, 2002.

[9] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Trans. Mob. Comput.*, 6(6):606–620, 2007.

[10] E. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *MobiHoc '07*, pages 32–40, 2007.

[11] M. S. Darko Kirovski, Nuria Oliver and D. Tan. Health-os: A position paper. In *ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments*, 2007.

[12] M. Demirbas, C. Rudra, A. Rudra, and M. A. Bayir. imap: Indirect measurement of air pollution with cellphones. *Technical Report, Deparment of Computer Science and Engineering, University at Buffalo*, September 2008.

[13] N. Eagle and A. Pentland. Social serendipity: Mobilizing social software. *IEEE Pervasive Computing*, 04-2:28–34, 2005.

[14] N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006.

[15] A. L. et al. Impact of communication infrastructure on forwarding in pocket switched networks. In *SIGCOMM '06*, pages 261–268, 2006.

[16] J. B. et al. Participatory sensing. In *ACM Sensys World Sensor Web Workshop*, 2006.

[17] T. A. et al. Mobiscopes for human spaces. *IEEE Pervasive Computing*, 6(2):20–29, 2007.

[18] M. Garetto and E. Leonardi. Analysis of random mobility models with pde's. In *MobiHoc*, pages 73–84, 2006.

[19] J. Ghosh, S. J. Philip, and C. Qiao. Sociological orbit aware location approximation and routing (solar) in manet. *Ad Hoc Networks*, 5(2):189–209, 2007.

[20] M. Gonzalez, C. Hidalgo, and A. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, 2008.

[21] J. Gulliver and D. Briggs. Time-space modeling of journeytime exposure to traffic-related air pollution using gis. *Environ. Res.*, 97(1):10–25, 2005.

[22] A. Harrington and V. Cahill. Route profiling: putting context to work. In *SAC*, pages 1567–1573, 2004.

[23] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden. Cartel: a distributed mobile sensor computing system. In *SenSys*, pages 125–138, 2006.

[24] S. H. K. L. I. Rhee, M. Shin and S. Chong. On the levy-walk nature of human mobility. In *In Proc. of IEEE INFOCOM*, 2008.

[25] W. jen Hsu, T. Spyropoulos, K. Psounis, and A. Helmy. Modeling time-variant user mobility in wireless mobile networks. In *INFOCOM*, pages 758–766, 2007.

[26] J. A. K. Sexton, S.J. Mongin and et al. Estimating volatile organic compound concentrations in selected microenvironments using time-activity and personal exposure data. *J. Toxicol. Environ. Health A.*, 70(5):465–476, 2007.

[27] A. Kansal, M. Goraczko, and F. Zhao. Building a sensor network of mobile phones. In *IPSN*, pages 547–548, 2007.

[28] A. Krause, E. Horvitz, A. Kansal, and F. Zhao. Toward community sensing. In *IPSN*, pages 481–492, 2008.

[29] K. Laasonen. Clustering and prediction of mobile user routes from cellular data. In *PKDD*, pages 569–576, 2005.

[30] K. Laasonen. Route prediction from cellular data. In *CAPS*, pages 147–158, 2005.

[31] J.-K. Lee and J. C. Hou. Modeling steady-state and transient behaviors of user mobility: : formulation, analysis, and application. In *MobiHoc*, pages 85–96, 2006.

[32] T. Liu, P. Bahl, S. Member, and I. Chlamtac. Mobility modeling, location tracking, and trajectory prediction in wireless atm networks. *IEEE Journal on Selected Areas in Communications*, 16:922–936, 1998.

[33] J. Markoulidakis, G. Lyberopoulos, D. Tsirkas, and E. Sykas. Mobility modeling in third-generation mobile telecommunication systems. *IEEE Personal Communications*, pages 41–56, August 1997.

[34] N. Marmasse and C. Schmandt. A user-centered location model. *Personal and Ubiquitous Computing*, 6(5/6):318–321, 2002.

[35] A. G. Miklas, K. K. Gollu, K. K. W. Chan, S. Saroiu, P. K. Gummadi, and E. de Lara. Exploiting social interactions in mobile systems. In *Ubicomp*, pages 409–428, 2007.

[36] C. M. Mirco Musolesi. Mobility models for systems evaluation a survey. Survey, 2008.

[37] P. Nurmi and J. Koolwaaij. Identifying meaningful locations. In *In Proc. 3rd Annual International Conference on Mobile and Ubiquitous Computing (Mobiquitous, Sun Jose, CA, USA, July 2006), IEEE Computer Society, 2006.*, 2006.

[38] N. Oliver and F. Flores-Mangas. Mptrain: a mobile, music and physiology-based personal trainer. In *Mobile HCI*, pages 21–28, 2006.

[39] A. Pentland. Automatic mapping and modeling of human networks. *Physica A: Statistical Mechanics and its Applications*, 378:41, 2006.

[40] L. Wang, Y. Jia, and W. Han. Instant message clustering based on extended vector space model. In *ISICA*, pages 435–443, 2007.

[41] M. Zonoozi and P. Dassanayake. User mobility modeling and characterization of mobility pattern. *IEEE J. Selec. Areas Commun.*, 15 (7):1239–1252, 1997.