Recognition of Periodic Behavioral Patterns from Streaming Mobility Data

Mitra Baratchi(✉), Nirvana Meratnia, and Paul J.M. Havinga

Department of Computer Science,
University of Twente, Enschede, The Netherlands
{m.baratchi,n.meratnia,p.j.m.havinga}@utwente.nl

Abstract. Ubiquitous location-aware sensing devices have facilitated collection of large volumes of mobility data streams from moving entities such as people and animals, among others. Extraction of various types of periodic behavioral patterns hidden in such large volume of mobility data helps in understanding the dynamics of activities, interactions, and life style of these moving entities. The ever-increasing growth in the volume and dimensionality of such Big Data on the one hand, and the resource constraints of the sensing devices on the other hand, have made not only high pattern recognition accuracy but also low complexity, low resource consumption, and real-timeness important requirements for recognition of patterns from mobility data. In this paper, we propose a method for extracting periodic behavioral patterns from streaming mobility data which fulfills all these requirements. Our experimental results on both synthetic and real data sets confirm superiority of our method compared with existing techniques.

1 Introduction

With ever-increasing emergence of ubiquitous location-aware sensing technologies, collecting huge volumes of mobility data streams from moving entities has nowadays become much easier than before. Mining and analyzing such large mobility data can uncover information about behaviors, habits, life style of moving entities, and their interaction [1]. Periodicity is an important essence of the activities of humans and animals. Animal's yearly migration and weekly work pattern of humans are examples of periodic behavioral patterns. Knowledge of such periodicity is required in various domains. For example, ecologists are interested to know the periodic migration pattern of animals and how human activities in vicinity of their living terrain cause abnormality in this behavior [2, 3]. In humanitarian studies, it is interesting to identify interruptions in periodic routines by major life events or daily hassles, as this identification helps in understanding stress-induced changes in daily behavior of people [4]. Identification of such abnormalities in human behavior can be useful in designing solutions which alleviate the effect of such stresses (as used in various healthcare based participatory sensing systems [5]).

Apart from uncertainties associated with mobility data (such as noise and missing samples) which make mining periodic patterns challenging, online extraction of patterns from streaming mobility data is difficult due to availability of limited processing and memory resources. The problem of identification of periodic behavioral patterns has been studied previously. What distinguishes this paper from the existing research,

I. Stojmenovic et al. (Eds.): MOBIQUITOUS 2013, LNICST 131, pp. 102–115, 2014.
DOI: 10.1007/978-3-319-11569-6_9

however, is its focus on identification of periodic patterns from *streaming mobility data* through a *light, accurate,* and *real-time* technique. Our automatic pattern recognition method requires limited storage and processing capability and is able to detect periodic patterns upon arrival of every new mobility measurement. To this end and in the context of identification of periodic patterns from streaming mobility data, our contributions in this paper are:

- *accurate discovery* of *periods* of repetitive patterns from streaming mobility data
- *real-time* extraction of *periodic patterns* with *bounded memory requirement*
- performance evaluation using both *synthetic* and *real data sets*.

The rest of this paper is organized as follows. Related work is presented in Sect. 2. In Sect. 3, we will define the problem of finding periodic patterns from streaming mobility data. Our methodology is described in detail in Sect. 4. Sections 5 and 6 present performance evaluation, and conclusions, respectively.

2 Related Work

Existing solutions for pattern mining from mobility data can be divided into solutions addressing either *frequent* pattern mining or *periodic* pattern mining. The former techniques focus on the "number of times" a pattern is repeated, while the latter focus on the "temporal trend by which" a pattern repeats itself.

Frequent Pattern Mining: Association rule mining [6] has been popularly used for extracting frequent trajectory patterns [7–11]. The general approach taken by all these techniques is to use a support-based mechanism to find the *longest frequent trajectory pattern*. Support-based mechanisms focus on the number of occurrences of patterns. The main drawback of exiting frequent pattern mining techniques is that the longest frequent pattern cannot completely and accurately describe the normal behavior. Specifically, these techniques fail to detect behaviors that do not occur frequently but they happen more than a prior expectation at a certain period.

Periodic Pattern Mining: In the domain of time series analysis there are a number of papers considering different questions regarding periodicity [12], such as asynchronous periodic patterns [13], and partial periodic patterns [14] of time series. Recently, mining periodic patterns from mobility data has also received attention [15–17]. The authors of [15] proposed an automatic periodicity detection mechanism to find the periodic behaviors. They further extended their work for extracting periodicity from incomplete observations in [17]. Similar to [17] we are interested in detection of periodic patterns from incomplete data. However, there are two main differences between the two techniques. Firstly, detection of periodic behavior in [17] is based on reference spots. These spots are places where the moving object spends a considerable amount of time. Therefore, it is needed that the regions of interest are extracted beforehand. This requires a preprocessing phase, which is not needed by our technique, as we work with raw GPS measurements. Secondly, method of [17] is not designed for streaming data and consumes considerable amount of memory. Our method, on the other hand, has low resource consumption and complexity which makes it applicable in streaming settings.

3 Problem Definition

In this section, we clearly define the problem of finding periodic patterns from streaming mobility data. We first start by providing some definitions:

Definition 1: A trajectory $L_1, L_2, \ldots$ is composed of a sequence of points denoted by $L_i = (x_i, y_i, t_i)$ where (x_i, y_i) represents a spatial coordinate and t_i is a time-stamp.

Definition 2: A period of length T is a time frame composed of T equally-sized segments denoted by $seg_{1..T}^{T}$.

Definition 3: A spatial neighborhood $sn_{(x_i,y_i)}$ is a set of all points that fall within the radius r of (x_i, y_i).

Definition 4: A spatial neighborhood is visited periodically in a period T, if the probability of being in this neighborhood in a seg_t^T of period T is more than a threshold in all or a fraction of observation time.

Problem: Having memory of size $6T_{\max}$ where $T_{\max}$ is our guess about the maximum period followed in data, we are interested in the latest periodic pattern followed in data stream $L_1 \ldots L_i (i > 6T_{\max})$ in form of $< T, \left[SN_1^T, \ldots, SN_T^T\right] >$ where T is a period and SN_t^T is either empty or it is a spatial neighborhood $sn_{(x_j,y_j)}$ which is expected to be visited periodically in seg_t^T.

4 Methodology

Our method to find periodic patterns from streaming mobility data is composed of three stages (shown in Fig. 1): (i) Measuring the self-similarity of the streaming data in different lags (described in Sect. 4.1), (ii) discovery of the periods of repetition from the self-similarity graph (described in Sect. 4.2), and (iii) extracting periodic patterns (described in Sect. 4.3).

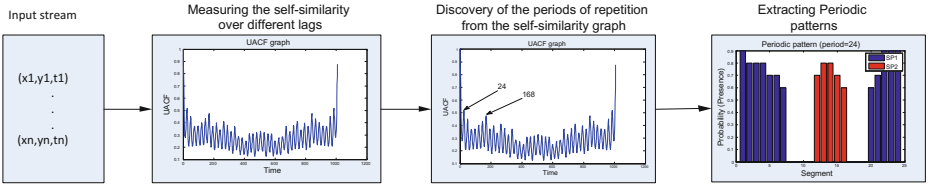

Fig. 1. Our framework for finding periodic patterns from streaming mobility data.

4.1 Measuring Self-Similarity of the Mobility Data in Different Lags

Behavioral patterns can have different periodicities (e.g. daily, weekly, monthly, and yearly). Therefore, it is important to be able to identify the period of repetition of visits

to a certain spatial neighborhood. One of the most commonly used methods[1] for identifying these periods is the circular Auto-Correlation Function (ACF) [18]. ACF measures the similarity of a time-series to itself in different lags. ACF of a time series ts, of size N over lags $\tau \in \{1...N\}$ is computed as follows:

$$ACF_N(\tau) = \sum_{i=1}^{N} ts(i).ts(i+\tau) \tag{1}$$

Due to difficulties such as cloud cover, or device malfunction, GPS data is often sparsely measured and mixed with noise while ACF requires the data to be uniformly sampled.

In order to measure the self-similarity from GPS measurements we propose the following optimization to the original ACF: Assuming that we denote missing samples with invalid and the rest with valid, we calculate the Uncertain circular Auto-Correlation Function (UACF) for a set of the mobility data $(L_1...L_N)$ using Eq. 2:

$$UACF_N(\tau) = \frac{1}{v^{\tau}_{1..N}} \sum_{i=1}^{N} \Psi_{i,i+\tau} \tag{2}$$

Where $\Psi_{i,i+\tau}$ is equal to 1 when the Euclidean distance between a valid pair L_i and $L_{i+\tau}(dist(L_i, L_{i+\tau}))$ is less than a threshold θ, and $v^{\tau}_{1..N}$ is the number of pairs $(i, i+\tau)$ in which both $L_i, L_{i+\tau}$ are valid. Computing UACF in this way will help us to measure the self-similarity of GPS data only in an offline fashion when the entire mobility data is available. In the next section, we optimize UACF (Eq. 2) to lower down its memory requirements and enable it to measure self-similarity over different lags upon arrival of each mobility data measurement.

4.1.1 Measuring Self-Similarity in Streaming Setting (Online)

We believe that finding periodic behavioral patterns in real-time helps in reducing the data transmission and storage (as not the raw data but only the patterns or whether the entity conforms to the pattern can be transmitted or stored). Computing UACF requires the entire data to be kept in memory. Therefore, its memory requirement is $O(N)$ (N is the number of measurements). Ubiquitous location-aware sensing devices have limited resources (both memory and power). Therefore, storing the entire data set (especially in case of high frequency sampled data set) for a long period of time or transmission of this data set to a central server for further analysis is neither practical nor possible. This motivates us to lower down the memory requirements. To do so, we need to calculate the UACF in such a way that upon arrival of each new GPS measurement L_N, we can still measure self-similarity over lags $\{\tau | N \bmod \tau = 0\}$. We claim that it is possible to reduce the memory requirement from $O(N)$ to $O(T_{\max})$, by having an estimation of the maximum period being followed in data ($T_{\max} \ll N$). (Since $N \bmod \tau = 0$ in what follows instead of N we use $n\tau$).

[1] Fourier transfrom is also used for period detection. However, this method has a low performance in identifying large periods [15].

Theorem. Suppose that $L_1 L_2 \ldots$ represent the stream of mobility data. We can compute the $\{UACF_{n\tau}(\tau) | \tau < T_{\max}\}$ for each $\{n > 3\}$ of this stream by having $O(T_{\max})$ memory.

Proof. In order to prove the above theorem we first prove that we can re-compute Eq. 2 through an alternative way. Consequently, we prove that in its new form, the memory requirement of computing UACF is bounded by $6*T_{\max}$. Therefore, we will first prove through mathematical induction that for each $(n > 3)$, $UACF_{n\tau}(\tau)$ can be computed as follows:

$$\begin{aligned} \mathrm{U}ACF_{n\tau}(\tau) = \frac{1}{v^{\tau}_{1..n\tau}} \Big(& v^{\tau}_{1..(n-1)\tau}\big(UACF_{(n-1)\tau}(\tau)\big) \\ & - \sum\nolimits_{i=1}^{\tau} \Psi_{(n-2)\tau+i,i} + \sum\nolimits_{i=1}^{\tau} \Psi_{(n-2)\tau+i,(n-1)\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{(n-1)\tau+i,i} \Big) \end{aligned} \tag{3}$$

Base Step. The base step is to check the validity of the above equation for $n = 4$. For $n = 4$ computing $UACF_{4\tau}(\tau)$ by Eq. 2 results in Eq. 4 and computing this value by Eq. 3 will result in Eq. 5 (please note that due to circular shift operation $\left(\sum_{i=1}^{\tau} \psi_{2\tau+i,3\tau+i} = \sum_{i=1}^{\tau} \psi_{2\tau+i,i}\right)$:

$$\begin{aligned} UACF_{4\tau}(\tau) &= \frac{1}{v^{\tau}_{1..4\tau}} \sum\nolimits_{i=1}^{4\tau} \Psi_{i,i+\tau} \\ &= \frac{1}{v^{\tau}_{1..4\tau}} \Big(\sum\nolimits_{i=1}^{\tau} \Psi_{i,\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{\tau+i,2\tau+i} \\ &\quad + \sum\nolimits_{i=1}^{\tau} \Psi_{2\tau+i,3\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{3\tau+i,i} \Big) \end{aligned} \tag{4}$$

$$\begin{aligned} UACF_{4\tau}(\tau) = \frac{1}{v^{\tau}_{1..4\tau}} \Big(& v^{\tau}_{1..3\tau}.(UACF_{3\tau}(\tau)) - \sum\nolimits_{i=1}^{\tau} \Psi_{i,2\tau+i} \\ & + \sum\nolimits_{i=1}^{\tau} \Psi_{2\tau+i,3\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{3\tau+i,i} \Big) \end{aligned} \tag{5}$$

We replace $UACF_{3\tau}(\tau)$ in Eq. 5 to see if it is equal to Eq. 4. Using Eq. 2 we will have:

$$\begin{aligned} UACF_{3\tau}(\tau) &= \frac{1}{v^{\tau}_{1..3\tau}} \sum\nolimits_{i=1}^{3\tau} \Psi_{i,i+\tau} \\ &= \frac{1}{v^{\tau}_{1..3\tau}} \Big(\sum\nolimits_{i=1}^{\tau} \Psi_{i,\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{\tau+i,2\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{2\tau+i,i} \Big) \end{aligned} \tag{6}$$

By replacing $UACF_{3\tau}(\tau)$ in Eq. 5 with Eq. 6 we get Eq. 4 as:

$$\begin{aligned} UACF_{4\tau}(\tau) &= \frac{1}{v^{\tau}_{1..4\tau}} \Big(v^{\tau}_{1..3\tau}.\Big(\frac{1}{v^{\tau}_{1..3\tau}}\Big)\Big(\sum\nolimits_{i=1}^{\tau} \Psi_{i,\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{\tau+i,2\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{2\tau+i,i} \Big) \Big) \\ &\quad - \sum\nolimits_{i=1}^{\tau} \Psi_{2\tau+i,i} + \sum\nolimits_{i=1}^{\tau} \Psi_{2\tau+i,3\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{3\tau+i,i} \Big) \\ &= \frac{1}{v^{\tau}_{1..4\tau}} \Big(\sum\nolimits_{i=1}^{\tau} \Psi_{i,\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{\tau+i,2\tau+i} \\ &\quad + \sum\nolimits_{i=1}^{\tau} \Psi_{2\tau+i,3\tau+i} + \sum\nolimits_{i=1}^{\tau} \Psi_{3\tau+i,i} \Big) \end{aligned} \tag{7}$$

Induction Step. Let $\{k \in \mathbb{N} | k > 3\}$ be given and assume Eq. 3 is true for $n = k$. Then we can prove that the Eq. 3 is valid for $n = k + 1$ as below:

$$
\begin{aligned}
UACF_{(k+1)\tau}(\tau) &= \frac{1}{v^{\tau}_{1..(k+1)\tau}} \sum_{i=1}^{(k+1)\tau} \Psi_{i,i+\tau} \\
&= \frac{1}{v^{\tau}_{1..(k+1)\tau}} \left(\sum_{i=\left(v^{\tau}_{1..k\tau} \not/ v^{\tau}_{1..k\tau}\right)1}^{\tau} \Psi_{i,\tau+i} + \cdots + \sum_{i=1}^{\tau} \Psi_{((k+1)-3)\tau+i,((k+1)-2)\tau+i} + \sum_{i=1}^{\tau} \Psi_{((k+1)-2)\tau+i,((k+1)-1)\tau+i} + \right. \\
&\left. \sum_{i=1}^{\tau} \Psi_{((k+1)-1)\tau+i,i}\right) \\
&= \frac{1}{v^{\tau}_{1..(k+1)\tau}} \left(\left(\sum_{i=1}^{\tau} \Psi_{i,\tau+i} + \cdots + \sum_{i=1}^{\tau} \Psi_{(k-2)\tau+i,(k-1)\tau+i}\right) + \sum_{i=1}^{\tau} \Psi_{((k+1)-2)\tau,((k+1)-1)\tau+i} + \sum_{i=1}^{\tau} \Psi_{((k+1)-1)\tau+i,i}\right) \\
&= \frac{1}{v^{\tau}_{1..(k+1)\tau}} \left(\left(v^{\tau}_{1..k\tau}\right) \left(\not{1} / v^{\tau}_{1..k\tau}\right) \left(\sum_{i=1}^{\tau} \Psi_{i,\tau+i} + \cdots + \sum_{i=1}^{\tau} \Psi_{(k-2)\tau+i,(k-1)\tau+i} + \sum_{i=1}^{\tau} \Psi_{(k-1)\tau+i,i}\right) - \right. \\
&\left. \sum_{i=1}^{\tau} \Psi_{(k-1)\tau+i,i} + \sum_{i=1}^{\tau} \Psi_{((k+1)-2)\tau,((k+1)-1)\tau+i} + \sum_{i=1}^{\tau} \Psi_{((k+1)-1)\tau+i,i}\right) \\
&= \frac{1}{v^{\tau}_{1..(k+1)\tau}} \left(v^{\tau}_{1..k\tau} \cdot (UACF_{k\tau}(\tau)) - \sum_{i=1}^{\tau} \Psi_{((k+1)-2)\tau+i,i} + \sum_{i=1}^{\tau} \Psi_{((k+1)-2)\tau,((k+1)-1)\tau+i} + \sum_{i=1}^{\tau} \Psi_{((k+1)-1)\tau+i,i}\right)
\end{aligned}
$$

Now we prove that we can calculate Eq. 3 with bounded memory. In this equation, $\sum_{i=1}^{\tau} \Psi_{(n-1)\tau+i,i}$ is calculated from $L_{1...\tau}$ and $L_{(n-1)\tau+1...n\tau} \cdot \sum_{i=1}^{\tau} \Psi_{(n-2)\tau+i,(n-1)\tau+i}$ is calculated from $L_{(n-2)\tau+1...n\tau}$. $UACF_{(n-1)\tau}(\tau)$ and $\left(\sum_{i=1}^{\tau} \Psi_{(n-2)\tau+i,i}\right)$ are single values computed in the previous round. It is straightforward with induction to prove that we can also compute $v^{\tau}_{1...n\tau}$ from $v^{\tau}_{1...(n-1)\tau}$ through $\left(v^{\tau}_{1...n\tau} = v^{\tau}_{1...(n-1)\tau} - v^{\tau}_{(n-2)\tau...\tau} + v^{\tau}_{(n-2)\tau...n\tau}\right)$ where $v^{\tau}_{(n-2)\tau...\tau}, v^{\tau}_{(n-2)\tau...n\tau}$ are computed from $L_{1...\tau}$ and $L_{(n-1)\tau+1...n\tau}$ (The proof is omitted due to lack of space). We know that $(\tau < T_{\max})$ so $(L_{1...\tau} L_{1...T_{\max}})$ and $\left(L_{(n-2)\tau+1...n\tau\cdot} \in L_{(n\tau-2T_{\max}+1)...n\tau}\right)$. Therefore, if we have $L_1 \ldots T_{\max}$, $L_{(n\tau-2T_{\max}+1)...n\tau}$ and $\{v^{\tau}_{1...n\tau}, UACF_{(n-1)\tau}(\tau), \sum_{i=1}^{\tau} \Psi_{i,(n-2)\tau+i} | \tau < T_{\max}\}$ in memory we can compute $UACF_{N=n\tau}(\tau)$ for any τ. Thereby, instead of keeping N measurements in memory we only need to keep $6T_{\max}(T_{\max} \ll N)$ values and the rest of data can be removed. As stated before, by having an estimation of $T_{\max}$, the correct periods can be extracted. In order to have the highest accuracy, choosing $T_{\max}$ can be performed considering the maximum memory available and changing the sampling rate.

4.2 Discovery of Periods of Repetition

If there is a single period of repetition in a time-series, the self-similarity graph (with both ACF and UACF) will show a peak in that period and all of its integer multiples. For instance, if there is a pattern repeated with period of 24 then the peaks will appear at 24, 48, 72, and so on. In order to extract periods of repetition from the self-similarity graph, normally the first highest peak is chosen. Since we cannot ignore the fact that there may exist multiple periodic patterns in mobility data, it is advantageous to be able to extract all periodic patterns and not only the one with the first highest peak. To clarify the case, in which multiple periodic patterns exist, let us consider the following example. Consider Bob, a student, who goes to school every weekday during the study year and stops going to school during summer. From one perspective, this behavior is periodic over a year (9 months going to school and 3 months holiday). From another view, we can also observe some other periods of repetition in this behavior (24 h, 7 days) as Bob goes to school every weekday and stops going to school on weekends. If we build a binary presence sequence for this activity of Bob for four years by

placing 1 at each time stamp when Bob is present at school and 0 at other times, the self-similarity graph by computing *ACF* on this sequence will look like Fig. 2(a, b).

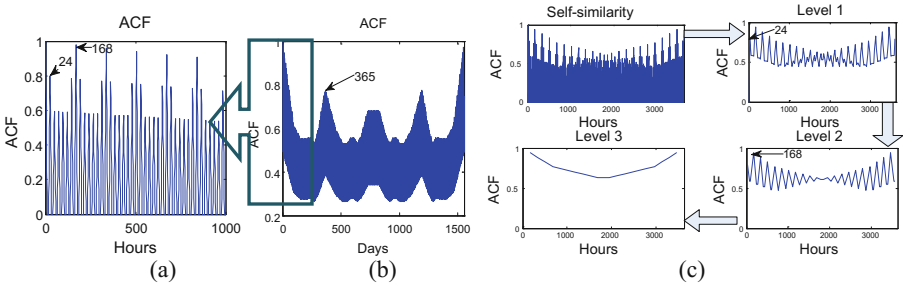

Fig. 2. (a) ACF self-similarity graph on the presence sequence of Bob on visiting school for the first 1000 h of 4 years ($\tau = 1$ h). (b)The result of performing ACF on the presence sequence data of Bob on visiting school ($\tau = 24$ h). (c) Extracting periods of repetition (Algorithm 1).

As seen in Fig. 2(a, b), in this self-similarity graph there are multiple valleys and hills, which are hierarchically ordered. The peaks with the highest ACF result are the ones which belong to the multiples of longer periods (in this example 365 days) and the lower hills belong to multiples of shorter periods (24 and 168). We can see intuitively in Fig. 2(c) that if we iteratively get peaks of self-similarity graph we can find such periods by choosing the first peak in each iteration. This will enable us to define periods of repetition as:

Definition 5: *Time lags $T_1 \ldots T_n$ are the periods of repetition in a data stream if (i) the self-similarity graph has a local maxima in lags $T_1 \ldots T_n$ and (ii) T_i is the first peak among peaks of level i−1 which is repeated in integer multiplies $(2T_i, 3T_{i,\ldots})$.*

Our procedure of extracting the periods of repetition is presented in Algorithm 1.

Algorithm 1: Extraction of Periods of Repetition

INPUT: $UACF_N(1 \ldots N)$(self-similarity graph)
OUTPUT: T (set of periods)

1: Find first level peaks, $Peaklevel(1)$ among $UACF(1 \ldots T)$ and set $i = 1$;
3: **Repeat while** $Peaklevel(i)$ is not empty
4: Find $Peaklevel(i+1)$ among $Peaklevel(i)$ and set $i = i + 1$;
5: **For each** $(j < i)$
6: Set period $T(j)$ to the first peak in $Peaklevel(j)$ which is repeated in integer multiplies;

4.3 Extracting Periodic Patterns in Streaming Setting

Successful discovery and extraction of periods of repetition only tells us that some spatial neighborhoods are visited periodically. This, however, does not indicate which spatial neighborhoods and when (in which segment of the period) they have been visited. Considering that the random existence of a moving entity in a spatial neighborhood $sn_{(x_j,y_j)}$ at seg_t^T of a discovered period T follows a Bernouli distribution (being in $sn_{(x_j,y_j)}(1)$, not being in $sn_{(x_j,y_j)}(0)$), the probability that this entity appears in $sn_{(x_j,y_j)}$

at seg_t^T randomly is 1/2. If this probability is more than 1/2, it shows that the moving entity has not appeared in that $sn_{(x_j,y_j)}$ randomly and its visit conforms to a periodic pattern. Therefore, in order to find the periodic patterns we need to find spatial neighborhoods which have been visited with a probability more than ½ in each segment of the discovered period of repetition. Algorithm 2 summarizes how we can extract both temporary and permanently periodic behaviors from streaming data. The algorithm proceeds as follows. Firstly, we use UACF to extract the periods. Next, for each discovered period of repetition T_i, we update the entries of a list of size T_i (referred to as $PL^{T_i}, PL^{T_i} = \left[\left(P_1^{Ti}, V_1^{Ti}, SN_1^{Ti}\right), \ldots, \left(P_{Ti}^{Ti}, V_{Ti}^{Ti}, SN_{Ti}^{Ti}\right)\right]$). For each spatial neighborhood $SN_i^{T_i}$, $P_i^{T_i}$ denotes the number of presences in $SN_i^{T_i}$ and $V_i^{T_i}$ represents the number of valid observations $V_i^{T_i}$ in segment $seg_i^{T_i}$. In each timestamp entities of PL^{T_i} lists get updated. Each measurement $\{L_N | N \bmod T_i = t\}$ will be compared with the value of $SN_t^{T_i}$ of PL^{T_i} list. In case the measurement lies within $2r$ from $SN_t^{T_i}$, the value of $SN_t^{T_i}$ will be updated with the average of the previous $SN_t^{T_i}$ values and the new value L_N. The values of $P_t^{T_i}$ and $V_t^{T_i}$ will be also updated correspondingly. Finally, the pattern composed of the value of spatial neighborhoods with a probability over (1/2) will be returned as periodic pattern and those $SN_t^{T_i}$ with a probability less than (1/2) will be removed.

Algorithm 2: Extraction of Periodic Patterns

INPUT: L_N(data point), *Buffer*, $PL^{T=1\ldots TMax}=[P_{i..T}^T, V_{i..T}^T, SN_{i..T}^T]$, T_{max}, r(radius)

OUTPUT: *Buffer*, $PL^{T=1\ldots TMax}=[P_{i..T}^T, V_{i..T}^T, SN_{i..T}^T]$, $PPatterns_{1\ldots T_{max}}$

1: Add L_N to the end of the *Buffer* and remove a point from the beginning of *Buffer;*
2: Update $UACF_N(\tau)$ using the *Buffer* where $N \bmod \tau = 0$;// Equation (3)
3: Find periods of repetition $T_{1\ldots k}$ from self-similarity graph $UACF(1 \ldots T_{max})$;// Algorithm 1
4: **For each** period T_i in periods $T_{1\ldots k}$
5: $t = N \bmod T_i$
6: **If** (dist (SN_t^T, L_N)<$2r$) , $P_t^{T_i} = P_t^{T_i}+1$, $SN_t^{T_i} = (P_t^{T_i}.SN_t^{T_i} + L_n)/(P_t^{T_i} + 1)$;
7:
8: **Else if** $(\frac{P_t^{T_i}}{V_t^{T_i}} < 1/2)$, $SN_t^{T_i} = L_N$, $P_t^{T_i} = 1, V_t^{T_i}$=0;
9:
10: $V_t^{T_i} = V_t^{T_i} + 1$;
11:
12: $PPattern_{T_i}$=$\{SN_{t\in 1..T_i}^{T_i} | P_t^{T_i} > 1 \;\&\; \frac{P_t^{T_i}}{V_t^{T_i}} > 1/2)\}$

5 Performance Evaluation

5.1 Complexity Analysis

In this section, we analyze the processing complexity and memory resources needed for extracting periodic patterns from streaming data of size N by Algorithm 2 assuming that the maximum repetitive period in the stream is less than $T_{\max}$. We compare our method with the method proposed in [17] and with the original ACF. It should be mentioned that ACF and [17] only measure self-similarity. Therefore, we only have to address their memory and processing power in this task. In our method, arrival of each new point, extracting repetition periods, and updating the *PL* lists have processing

complexity of $(T_{\max})$, $O(T_{\max} \log T_{\max})$, and $O(T_{\max}^2)$, respectively. As shown in Sect. 4.1.1, we reduced the memory requirements of measuring self-similarity to $O(T_{\max})$ and discovery of the periods of repetition has memory complexity of $O(T_{\max})$. In pattern extraction, we keep a *PL* list of size T for each period $(T < T_{\max})$. Therefore, memory requirement of this task is $O(T_{\max}^2)$. The method proposed in [17] extracts periodicities from each region of interest (rather than original data). In order to perform real-time and streaming period extraction, this method should be able to identify the regions of interest first. The regions of interest are not known beforehand. Therefore, to be able to compare our technique with [17], we simply assume that we compare each new GPS measurement with cells of a grid of size G. In this case, the processing complexity for this comparison will be $O(G)$. In order to measure the self-similarity, this method requires having all the previous points in memory and update probability of presence in each segment of each period. Then it measures the self-similarity for each possible period by $O(T_{\max}N)$ processing. This task should be performed C number of times (C is a constant value) in order to normalize the data. Therefore, the processing power is $O(CNT_{\max}) + O(G)$ and memory requirements will be $O(N)$. Complexity of ACF using Eq. 1 is $O(N^2)$ and it also requires the whole data in memory. Table 1 summarizes the memory and processing complexity of these three techniques. As seen, only our method is suitable for streaming settings.

Table 1. Complexity comparison

Method	Processing			Memory		
	Measuring self-similarity	Period extraction	Pattern extraction	Period extraction	Period extraction	Pattern extraction
Our method	$O(T_{\max})$	$O(T_{\max} \log T_{\max})$	$O(T_{\max}^2)$	$O(T_{\max})$	$O(T_{\max})$	$O(T_{\max}^2)$
[17]	$O(G)$ $+O(GNT_{\max})$	–	–	$O(N)$	–	–
ACF	$O(N^2)$	–	–	$O(N)$	–	–

5.2 Performance Evaluation Using Synthetic Dataset

5.2.1 Synthetic Dataset

Validation with a synthetic dataset helps us to check the sensitivity of our period detection algorithm under several parameters which cause imperfections in mobility data. We wrote a moving object sequence generator to produce a synthetic periodic sequence of a person's movement in N number of days. This periodic sequence is in form of $test_i = \{(x_i, y_i) | i \in [1, N \times 24]\}$ where each index represents a spatial neighborhood where a person is between $[(i-1) \bmod 24, i \bmod 24]$ on the $(\frac{i}{24} + 1)$ th day. Ten spatial neighborhoods are defined, each composed of two dimensional points lying within radius r from a predefined center. We consider two of these spatial neighborhoods (representing home and office) being periodically visited (daily, and weekly) in specific intervals. For workdays, the interval 10:00-18:00 is chosen for "being at work" and 20:00-8:00 for "being at home". On weekends, the interval between 01:00-24:00 is chosen for "being at home". Each of these intervals is subject to a random event with

probability of μ and is normal otherwise. In normal intervals with defined start (t_{start}) and end (t_{end}), the event of "visit" (being at home or office) starts somewhere between $(t_{start} \pm \sigma_1)$ and ends around $(t_{end} \pm \sigma_2)$. The behavior in abnormal intervals is randomly chosen from other 9 spatial neighborhoods with a random start-time and random duration. Such abnormal intervals can represent different un-periodic events such as absence at work, working overtime, or visit to places such as cinemas, shops, etc. After defining the normally and abnormally visited places (spatial neighborhoods) for each day, we add trajectories between them, each with different duration. This can represent different modes of transport (for instance, car, or bike). The effect of missing samples was tested by removing data from the random indexes with probability of α. In order to add noise, we formed a randomly permuted array of data between the maximum and minimum longitude and latitudes in selected spatial neighborhoods. Next, we randomly picked indexes with probability of β and replaced them with the values in the random array. The parameters used to form the test sequence are: radius of spatial neighborhood (r = 100 m), number of periodic repetition (N = 100), missing samples (α=0–50 %), noise (β = 0–50 %), standard deviation of start/end-time $(\sigma_1, \sigma_2 = 2)$, and probability of random events (μ = 0–50 %).

5.2.2 Performance Evaluation with the Synthetic Dataset

The synthetic dataset generated by movement generator entails two periods of repetition (24, and 168 h corresponding to a day and a week). In this section, we evaluate Algorithm 1 to see how successful we are in extraction of these two periods using ACF and UACF self-similarity graphs (method of [17] is not applicable on raw data). We calculate self-similarity in different lags by ACF on latitude (lat), longitude (long) and their root mean square (RMS) $\sqrt{lat^2 + long^2}$. We test the effect of noise (β), missing samples (α), and random events (μ) on detection of correct periods by running the experiments 100 times (Fig. 3(a–f)). Figure 3(g) compares the precision computed by $\frac{P^+}{P^+ + P^-}$ where P^+ is the sum of correct prediction of two periods and P^- is the number of false alarms in all the previous experiments.

Looking at Fig. 3, we can see that UACF clearly outperforms ACF in presence of noise, missing samples and random events. Even when these parameters is near 50 %, considerably high percentage of correct periods is discoverable through using UACF by overcoming the effect of pattern-less data through taking into account the effect of points that fall into a spatial neighborhood. ACF, however, measures self-similarity by multiplying pattern-less data and those which follow a pattern. The overall precision using UACF is also higher than ACF.

5.3 Performance Evaluation Using Real Dataset

5.3.1 The Real Dataset

The real dataset we use (plotted in Figs. 4a and 5a), was collected using custom-designed GPS-enabled wireless sensor nodes carried around by two researchers. The devices were set to take one measurement per minute for a period of 31 days by first candidate and 109 days by the second one. When used inside the building, the nodes were placed near the window to obtain data. This however had made the dataset

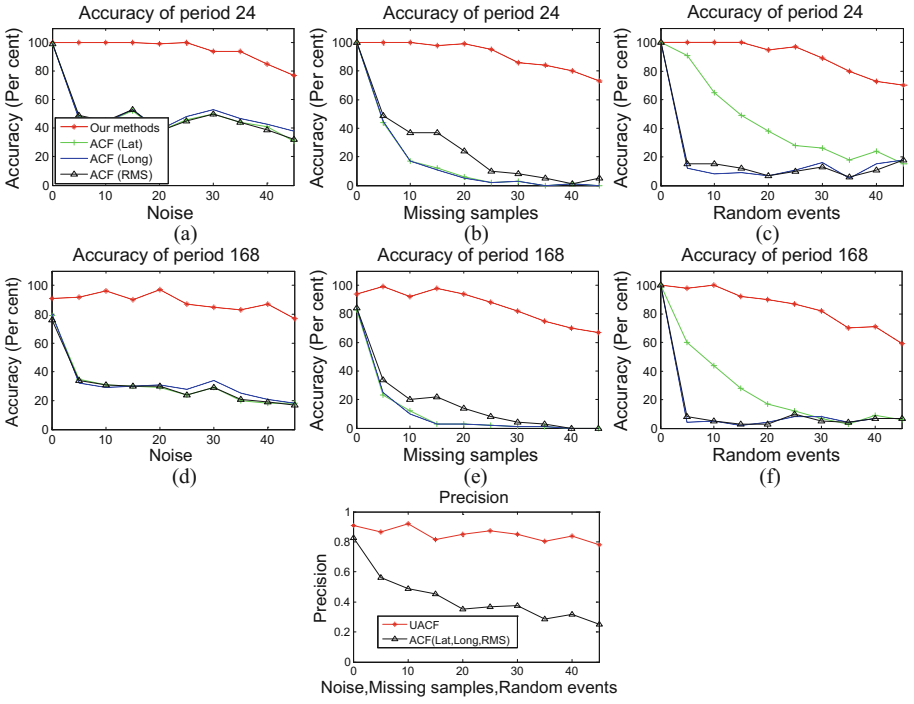

Fig. 3. (a–f) Comparison of the accuracy of Algorithm 1 in extracting periods of repetition (24,168) using UACF and ACF in presence of noise, missing samples and random events. (g) Average precision of Algorithm 1 in extracting periods of repetition.

extremely noisy. The data collected by first candidate is extremely sparse. This person, has kept the node off for all the weekends and the rest of data partly shows his regular behavior in commuting between home and work (weekdays) and very few irregular visit. The data collected by second candidate has less missing samples, while this person had a more dynamic behavior. She has gone on (i) work days to office, (ii) Saturdays to the open market in the city center, and (iii) regularly to a language class for a short period of time, and (iv) irregularly to a supermarket and a gym. Several other irregular behaviors have emerged for this person during the short period, such as traveling to another city, being absent at work or working overtime.

5.3.2 Performance Evaluation

Using the real data set, we calculated the self-similarity over different time lags with UACF and ACF (root mean square) (shown in Figs. 4b–c, 5b–c). We used Algorithm 1 to extract the periods of repetition from the self-similarity graph for both candidates. For the first candidate, we were able to extract the period of 24 h using UACF, while no period was found using ACF. We noticed that it was not possible to extract the period of 168 as no data was available for weekends. For the second candidate, UACF was able to

detect both periods of 24 and 168 h, while ACF could only find the period of 24. This is because as it can be seen in Fig. 4b–c, the lag of 24 has the first highest peak in ACF graph and there is no distinguishable peak after that. The hierarchy of peaks, however, is clearly distinguishable using UACF. Therefore, both periods were easily found using Algorithm 1. After finding the spatial neighborhoods for each segment of discovered periods using Algorithm 2, we merged those ones which were closer than the diameter of the spatial neighborhood. Our approach is able to find two spatial neighborhoods for the first candidate (his home and office) (Fig. 4a) and 3 spatial neighborhoods are identified for the second candidate (her home, office, and city center) (Fig. 5a).

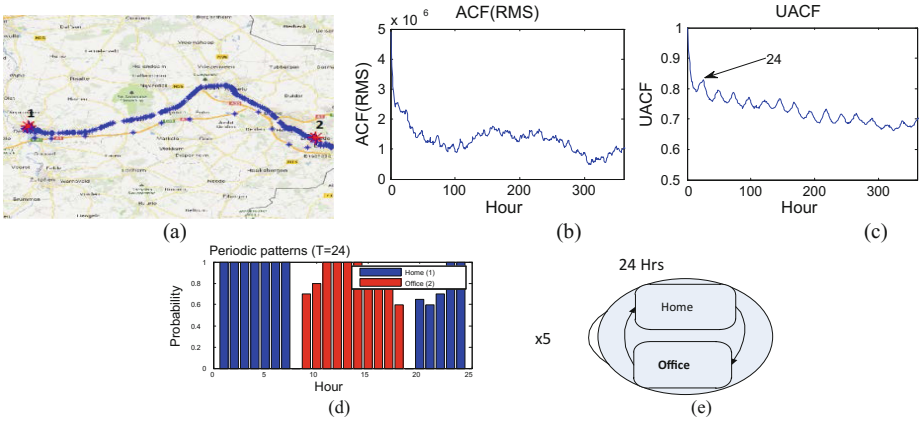

Fig. 4. (a) Mobility data stream (shown in blue) and identified periodically visited spatial neighborhood corresponding to this dataset (shown in red) of candidate 1. (b, c) Extracting periods from self-similarity graph of real dataset using ACF and UACF. (d) Periodic patterns extracted from algorithm 2, (e) state-diagram of periodic behavior.

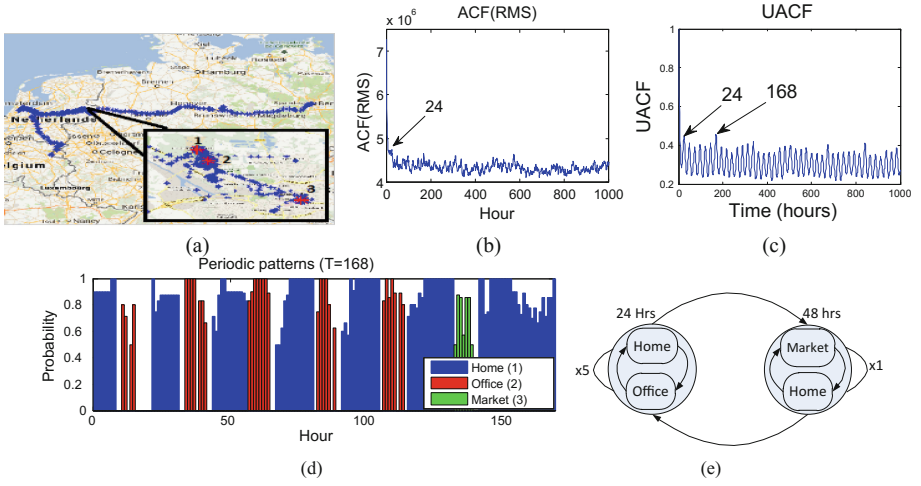

Fig. 5. Extracting periodic behavior of candidate 2. ((a–e) The same as Fig. 4)

The histograms in Figs. 4d and 5d are representing the probability of appearance in SP_i^T in segment seg_i^T of each of the larger discovered period found (from Algorithm 2). The state diagrams on right are drawn based on the histograms to represent the periodic pattern. As illustrated in the state diagrams, the periodic pattern of the first candidate is composed of a loop between home and work. For the second candidate, a periodic pattern of two loops is identified. These loops are repeated 5 times with the duration of 24 h (Weekdays). Next, a new loop of 48 h emerges which is only followed once, after which the first loop is repeated again.

6 Conclusion

In this paper, we address the problem of accurate and real-time extraction of periodic behavioral patterns from streaming mobility data using resource constrained sensing devices. We propose a method to identify correct periods, in which periodic behaviors occur from raw streaming GPS measurements. We then use these periods to extract periodic patterns. We empirically evaluated the performance of our method using a synthetic data set under different controllable parameters such as noise, missing samples, and random events. We also tested our technique on a real data set collected by two people. Results of our evaluations on both synthetic and real data sets show superiority of our technique compared to the existing techniques. In our future work, we plan to (i) test our technique for real data set of a large group of people and (ii) finding "abnormal" behaviors using streams of mobility data.

References

1. Baratchi, M., Meratnia, N., Havinga, P.J.M.: On the use of mobility data for discovery and description of social ties. In: Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013), Niagara Falls, Canada (2013)
2. Wisdom, M.J., et al.: Spatial partitioning by mule deer and elk in relation to traffic. In: Transactions of the 69th North American Wildlife and Natural Resources Conference, pp. 509–530 (2004)
3. Baratchi, M., et al.: Sensing solutions for collecting spatio-temporal data for wildlife monitoring applications: a review. Sensors **13**, 6054–6088 (2013)
4. Monroe, S.: Major and minor life events as predictors of psychological distress: Further issues and findings. J. Behav. Med. **6**, 189–205 (1983). 1983/06/01
5. Aflaki, S., et al.: Evaluation of incentives for body area network-based HealthCare systems. In: Proceedings of IEEE ISSNIP, Melbourne, Australia, (2013)
6. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., USA (1993)
7. Verhein, Florian, Chawla, Sanjay: Mining spatio-temporal association rules, sources, sinks, stationary regions and thoroughfares in object mobility databases. In: Li Lee, Mong, Tan, Kian-Lee, Wuwongse, Vilas (eds.) DASFAA 2006. LNCS, vol. 3882, pp. 187–201. Springer, Heidelberg (2006)

8. Giannotti, F., et al.: Trajectory pattern mining. In: Proceedings of 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA (2007)
9. Wei, L.-Y., Zheng, Y., Peng, W.-C.: Constructing popular routes from uncertain trajectories. In: Proceedings of 18th ACM SIGKDD, Beijing, China (2012)
10. Mamoulis, N., et al.: Mining, indexing, and querying historical spatiotemporal data. In: Proceedings of tenth ACM SIGKDD, Seattle, WA, USA (2004)
11. Baratchi, M., Meratnia, N., Havinga, P.J.M.: Finding frequently visited paths: dealing with the uncertainty of spatio-temporal mobility data. In: Proceedings of IEEE ISSNIP, Melbourne, Australia (2013)
12. Elfeky, M.G., Aref, W.G., Elmagarmid, A.K.: Periodicity detection in time series databases. IEEE Trans. Knowl. Data Eng. **17**, 875–887 (2005)
13. Jiong, Y., Wei, W., Yu, P.S.: Mining asynchronous periodic patterns in time series data. IEEE Trans. Knowl. Data Eng. **15**, 613–628 (2003)
14. Yang, R., Wang, W., Yu, P.S.: InfoMiner + : mining partial periodic patterns with gap penalties. In: Proceedings of ICDM 2002, pp. 725–728 (2002)
15. Li, Z., Ding, B., Han, J., Kays, R., Nye, P.: Mining periodic behaviors for moving objects. In: Proceedings of 16th ACM SIGKDD, Washington, DC, USA (2010)
16. Sadilek, A., Krumm, J.: Far Out: predicting long-term human mobility. In: Proceedings of Twenty-Sixth AAAI Conference on Artificial Intelligence, pp. 814–820 (2012)
17. Li, Z., Wang, J., Han, J.: Mining event periodicity from incomplete observations. In: Proceedings of 18th ACM SIGKDD, Beijing, China, (2012)
18. Oppenheim, A.V., Schafer, R.W., Buck, J.R.: Discrete-Time Signal Processing. Prentice Hall, Upper Saddler River, NJ (1999)