

# Semi-Markov State Estimation and Policy Optimization for Energy Efficient Mobile Sensing

Yi Wang, Bhaskar Krishnamachari, Murali Annavaram

Viterbi School of Engineering, University of Southern California, Los Angeles, CA 90089, USA

{wangyi,bkrishna,annavara}@usc.edu

**Abstract**—User/environmental context detection on mobile devices benefits end-users by providing information support to various kinds of applications. A pervasive question, however, is how the sensors on the mobile device should be sampled energy efficiently without sacrificing too much detection accuracy. In this paper, we formulate the user state sensing problem as the intermittent sampling of a semi-Markov process, a model that provides general and flexible capturing of realistic data with any type of state sojourn distributions. We propose (a) a semi-Markov state estimation mechanism that selects the most likely user state while observations are missing, and (b) a semi-Markov optimal sensing policy  $u_s^*$  which minimizes the expected state estimation error while maintaining a given energy budget. Their performance are shown to significantly outperform Markov algorithms on simulated two-state processes and real user state traces pertaining to different types of state distributions. Finally, in order to evaluate the performance of  $u_s^*$ , we implement a client-server based basic human activity recognition system on N95 smartphones and desktops which automatically computes user-specific optimal sensing policy based on historically collected data. We show that  $u_s^*$  improves the estimation accuracy by 27.8% and 48.6% respectively over Markov-optimal policy and uniform sampling through a set of experiments.

## I. INTRODUCTION

Current generation smartphones integrate a wide range of sensing and networking features such as GPS, accelerometer, Bluetooth, and WiFi, which are able to recognize users' context and surrounding information unobtrusively and in real time. By sampling user/environmental context, the mobile clients are able to better support high level applications such as health monitoring, location/activity based services, and content distributions. For example, it would be much more convenient if mobile phones could automatically adjust the ring tone profile according to the surroundings and the events in which the users are participating; in mobile p2p networking applications, performing a scan to identify the number and link quality of peering devices may help the mobile device make better decisions on efficient and reliable data transmissions.

In our study, we use the term “state” to represent (discretized) user/environmental context, which could possibly represent users’ activity, location, connectivity (to peer/infrastructure node), background information, and so on. A critical performance trade-off lies between the accurate detection of user state and the energy consumption spent by sensor samplings. On the one hand, most applications require that user state is correctly recognized and estimated; on the other hand, detecting user state requires sensor activations that consume significant amount of energy which may drain the

mobile device battery quickly. Although sensor duty cycling might be applied to reduce energy consumption, there exist two major problems that need to be carefully investigated: (1) how to accurately estimate the underlying state when the sensor stays idle? and (2) given an energy budget, how should sensor sampling be scheduled intelligently such that the expected state estimation error can be minimized?

To address the above problems, in this paper, we formulate the state sensing problem as the intermittent sampling of a semi-Markov process, a class of stochastic processes that allows any type of state sojourn time distribution and therefore provides more accurate capture of realistic data than the commonly used Markov process model. In particular, this paper makes the following contributions:

First, for an intermittently sampled semi-Markov process with truncated sojourn times, we propose a forward-backward approach in order to estimate the most likely state information whenever observations are missing. This real-time semi-Markov estimation mechanism requires neither the origin information nor the initial state distribution, since it functions by estimating the past sojourn time and the expected state ending time for each individual estimation interval. We demonstrate that it outperforms Markov estimation significantly on both simulated state sequences as well as real data traces.

Second, we utilize linear programming (LP) with specific definitions of costs (i.e., energy and error) to solve the constrained optimization problem in order to obtain the semi-Markov optimal sensing policy  $u_s^*$  that minimizes expected state estimation error while satisfying an energy constraint. We show that  $u_s^*$  achieves great performance improvement over the Markov-optimal policy  $u_m^*$  [33] on real data traces.

Third, we design and implement a client-server based energy efficient mobile sensing system in order to evaluate the performance of semi-Markov sensing policy in real systems. The system is implemented on Nokia N95 devices (and desktop PC as back-end server) for basic human activity recognition which automatically computes and applies the user-specific optimal sensing policy based on previously collected user behavior dynamics. We demonstrate through a set of experiments that the semi-Markov optimal policy achieves the best estimation accuracy and longest battery lifetime as compared to Markov-optimal policy and uniform sampling.

## II. RELATED WORK

As the limited battery capacity often drives the design consideration in mobile computing, energy efficient mobile

sensing [17], [34], [19], networking [9], [26], [1], and software design [8], [6] have been widely investigated by researchers in recent years. For example, in the field of mobile sensing, Krause *et al.* [19] investigate the trade-off between prediction accuracy and power consumption in mobile computing. The design of SeeMon [17] and EEMSS [34] systems explore hierarchical sensor management and sensor triggering mechanisms in order to achieve energy efficient context detection. Driven by increasing user demands, energy efficient context detection has been investigated and applied in applications such as wearable health monitoring ([29], [35], [36], [4]) and localization ([23], [18], [16], [20], [39]).

Modeling context with discrete state space and stochastic processes (especially Markov processes) has been used in different types of mobile sensing applications. For example, Thiagarajan *et al.* [31] model the transition of user location (in terms of road segments) as a Markov chain and provide location estimation using HMMs in intelligent transportation systems. Thatte *et al.* [30] treat fine-grained user activity transition (e.g., sitting, walking, running, etc.) as a Markov chain and design sensor selection algorithms in order to provide high-fidelity obesity monitoring. Although not explicitly using a discrete state model, Wang *et al.* [32] investigated the distribution of contact arrivals in mobile p2p networks and studied its impact on energy efficient contact probing mechanisms.

In this paper, however, as strictly Markovian characteristics are rarely seen in real state traces, we formulate the user context detection problem as the intermittent sampling of a semi-Markov process. To reconstruct the user state process based on discontiguous observations, we propose a semi-Markov state estimation algorithm that follows the conventional forward-backward approach introduced by Ferguson in [12]. However, the traditional algorithm introduced in [12] deals with full observation sequence and hence cannot be directly applied to cases where sensors are mostly idle with sparse observations.

Yu and Kobayashi [38] have studied the hidden semi-Markov model (HSMM) with missing observations for mobility tracking applications. Their problem formulation assumes that the HSMM starts at the first observation and ends at the last one, and state estimation is conducted off-line for the entire process. In contrast, in this paper, we provide a generic estimation mechanism that can be executed in real time for any estimation interval based only on neighboring observations.

Stochastic optimization tools have been widely used to help design better protocols in resource-aware mobile operations. For example, Benini *et al.* [3] addressed power optimization on battery-operated devices with the assistance of Markov Decision Process (MDP). Cheung *et al.* [6] propose a MDP based optimization framework to optimize resource usage (e.g., battery-life), such that user-specific reward can be maximized. In our previous study [33], a Constrained Markov Decision Process (CMDP) based optimization framework has been proposed that schedules sensors intelligently while assuming a Markov user state process. In this paper, inspired by the standard use of LP to solve constrained

semi-Markov decision process (CSMDP) problem [10], we build a constrained optimization framework and develop its corresponding LP for intermittently sampled semi-Markov processes in order to find the best semi-Markov sensing policy that minimizes the expected estimation error under energy consumption constraint.

### III. SEMI-MARKOV STATE ESTIMATION WITH SOJOURN TIME TRUNCATION

#### A. Preliminaries

We assume that time is discretized and the user state evolves as a  $N$ -state discrete time semi-Markov process. Sensor observations are assumed to be perfect, and due to energy consumption concern, the sensor adopts duty cycling which leads to a sparsely sampled state sequence with multiple intervals of missing observations (see figure 1).

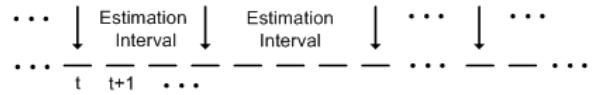


Fig. 1. The semi-Markov process is sampled at discontiguous time slots, leading to multiple estimation intervals.

Let  $O_t$  represent the observed state at time  $t$ , where  $O_t \in \{1, 2, \dots, N\}$ . Let  $p_{ij}$  denote the probability of user state transition from state  $i$  to state  $j$ . Similar to Markov models, the transition probabilities satisfy

$$p_{ij} \geq 0, \text{ and } \sum_{j=1}^N p_{ij} = 1, \forall i, j \in \{1, 2, \dots, N\}, \quad (1)$$

with no self-transitions, i.e.:

$$p_{ii} = 0, \forall i \in \{1, 2, \dots, N\}. \quad (2)$$

We further let  $w_i(a)$  ( $a \in \mathbb{N}^*$ ) denote the pmf of sojourn time of state  $i$ . It is assumed that  $1 \leq a \leq A_i < \infty$ , and  $\sum_a w_i(a) = 1$ , where  $A_i$  is the maximum sojourn duration of state  $i$ . For practical purposes, we approximate distributions with infinite support with a sufficiently large choice of  $A_i$ .

The following state estimation problem is investigated: given the state distribution parameters  $p_{ij}$  and  $w_i(a)$ , and two state observations  $O_{t_m} = o_1$  and  $O_{t_n} = o_2$ , what is the most likely state at any given time slot between  $t_m$  and  $t_n$ ?

It is important to note that, unlike the assumptions made in [12], [38], the observed states  $o_1$  and  $o_2$  do not necessarily need to start at time  $t_m$  or end at time  $t_n$ . Instead, the state estimation is conducted in real-time for any given estimation interval, where sensor observations could be made at arbitrary time slots, and the state detection may belong to any instance of the sojourn time duration.

#### B. The Forward-backward variables

As the state observation  $o_1$  may truncate the state sojourn duration, we define  $t'_m$  (where  $t_m - A_{o_1} < t'_m \leq t_m$ ) as the "true" time at which state  $o_1$  is entered, i.e., the observed state  $o_1$  started its sojourn time at time  $t'_m$ , and lasted through time  $t_m$ . In real-time system implementation where state durations are probabilistic, the exact value of  $t'_m$  is often unavailable

since the sensor makes sparse observations and may not have the knowledge of a complete historic state sequence. Therefore, for a state evolving process that is divided into multiple estimation intervals,  $t'_m$  could be obtained through the following approximation alternatives:

Method 1 (**M1**): Identify the starting time of  $o_1$  using the state estimation result from the previous estimation interval.

Method 2 (**M2**): Use the expected value  $E_p$  of past sojourn time, which can be calculated as follows (for state  $i$ ):

$$E_p(i) = \sum_{a=1}^{A_i} \sum_{\tau=1}^a \frac{w_i(a) \cdot \tau}{a}. \quad (3)$$

Method 3 (**M3**): Set  $t'_m = t_m$ , i.e., it is assumed the state sojourn time starts at the beginning of each observation interval. This assumption is implicitly used in [12] and [38], however, in section III-D we will demonstrate that this method leads to the most undesirable performance among all options.

Since M1 conducts state estimation based on previously estimated results, it may cause error propagation where previous estimation error may accumulate and lead to undesired performance in future estimations. M2 and M3 do not suffer from this problem because they approximate the history of state sojourn durations using expected and zero past sojourn time, respectively. The comparison of these three methods on simulated state sequences with different state sojourn distributions is presented in section III-D.

Let  $A$  be the maximum length of state sojourn times, i.e.:

$$A = \max_i \{A_i\}, \forall i \in \{1, 2, \dots, N\}. \quad (4)$$

For a given time slot  $t$ , we define forward variables  $f_t(i)$  and  $f_t^*(i)$  as:

$$\begin{aligned} f_t(i) &= \text{Prob}[o_1, t'_m, \text{state } i \text{ ends at } t] \\ &= \sum_{a=1}^A f_{t-a}(i) \cdot w_i^\Delta(a), \end{aligned} \quad (5)$$

where

$$w_i^\Delta(a) = \begin{cases} w_i(a), & \text{if } t - a \geq t_m, \\ w_i'(a) = \frac{w_i(a)}{\sum_{k=t_m-t'_m+1}^{A_i} w_i(k)}, & \text{else.} \end{cases} \quad (6)$$

and

$$\begin{aligned} f_t^*(i) &= \text{Prob}[o_1, t'_m, \text{state } i \text{ starts at } t+1] \\ &= \sum_{j=1}^N f_t(j) \cdot p_{ji}. \end{aligned} \quad (7)$$

The reason that  $w_i^\Delta(a)$  is defined differently from  $w_i(a)$  under the condition that  $t - a < t_m$ , is that the observed state at time  $t_m$  has already spent its sojourn time for  $t_m - t'_m + 1$  consecutive time slots such that the rest of the sojourn time distribution needs to be conditioned upon this fact, i.e.,  $w_i'(a) = \text{Prob}[\text{state } i \text{ lasts for } a \text{ slots} \mid a \geq t_m - t'_m + 1]$ .

In order to compute  $f_t(i)$ , the boundary condition for  $f_t^*(i)$  needs to be initialized as follows:

$$f_t^*(i) = \begin{cases} 1, & \text{if } t = t'_m - 1, \text{ and } i = o_1, \\ 0, & \text{otherwise.} \end{cases} \quad (8)$$

and

$$f_{t_m}^*(i) = p_{o_1 i} \cdot w_{o_1}(t_m - t'_m + 1), \forall i \neq o_1. \quad (9)$$

Equation (8) emphasizes the fact that the detected state  $o_1$  has started its sojourn time at time  $t'_m$  whereas equation (9) computes the probability that state  $o_1$  transits to another state in the first time slot of the estimation interval (i.e.,  $t_m + 1$ ).

To initialize  $f_t(i)$  at time  $t_m$ , the probability that observation  $o_1$  ends exactly at  $t_m$  needs to be calculated, which satisfies:

$$f_{t_m}(i) = \begin{cases} w_{o_1}'(t_m - t'_m + 1), & \text{if } i = o_1, \\ 0, & \forall i \neq o_1. \end{cases} \quad (10)$$

We further define the backward variables  $b_t(i)$  and  $b_t^*(i)$  for a given time slot  $t$ :

$$\begin{aligned} b_t(i) &= \text{Prob}[o_2 \mid \text{state } i \text{ starts at } t] \\ &= \sum_{a=1}^A b_{t+a}^*(i) \cdot w_i(a), \end{aligned} \quad (11)$$

and

$$\begin{aligned} b_t^*(i) &= \text{Prob}[o_2 \mid \text{state } i \text{ ends at } t-1] \\ &= \sum_{j=1}^N b_t(j) \cdot p_{ij}. \end{aligned} \quad (12)$$

Since  $o_2$  does not necessarily end at  $t_n$ , the initial and boundary conditions for  $b_t^*(i)$  are therefore given by:

$$b_{t_n}^*(o_2) = 0, \quad (13)$$

and

$$b_{t_n+x}^*(i) = \begin{cases} 1, \forall x \in \{1, 2, \dots, A_{o_2}\}, & \text{if } i = o_2, \\ 0, & \text{otherwise.} \end{cases} \quad (14)$$

Equation (14) describes that given the fact that state  $o_2$  ends at any time slot after time  $t_n$ , the probability of seeing  $o_2$  at time  $t_n$  is 1, and correspondingly, the probability of seeing other states at  $t_n$  is 0.

The initial conditions for  $b_t(i)$  need to be specified as:

$$b_{t_n}(i) = \begin{cases} \sum_{x=1}^{A_{o_2}} b_{t_n+x}^*(o_2) \cdot w_{o_2}(x), & \text{if } i = o_2, \\ 0, & \forall i \neq o_2. \end{cases} \quad (15)$$

in which  $b_{t_n}(o_2)$  is the average aggregated probability that state  $o_2$  begins at time  $t_n$ .

### C. State estimation

Once the values of all the forward and backward variables have been obtained by recursion, it is feasible to estimate the state for each time slot between  $t_m$  and  $t_n$ . We define a new forward variable:

$$\phi_t(i) = \text{Prob}[o_1, t'_m, s_t = i], \quad (16)$$

which holds the following property:

$$\begin{aligned} \phi_t(i) &= \text{Prob}[o_1, t'_m, s_{t-1} = i] \\ &\quad - \text{Prob}[o_1, t'_m, s_{t-1} = i \text{ and } s_t \neq i] \\ &\quad + \text{Prob}[o_1, t'_m, s_{t-1} \neq i \text{ and } s_t = i] \\ &= \phi_{t-1}(i) - f_{t-1}(i) + f_{t-1}^*(i). \end{aligned} \quad (17)$$

Therefore,  $\phi_t(i)$  can be calculated by recursion as well. To initialize, the following equalities hold:

$$\phi_{t_m}(i) = \begin{cases} 1, & \text{if } i = o_1, \\ 0, & \text{else.} \end{cases} \quad (18)$$

Next, we define a new backward variable:

$$\psi_t(i) = \text{Prob}[o_1, t'_m, o_2, s_t = i], \quad (19)$$

Similar to the relationship illustrated in equation(17),  $\psi_t(i)$  can be further written in the following recursive form:

$$\psi_t(i) = \psi_{t+1}(i) - b_{t+1}(i) \cdot f_t^*(i) + f_t(i) \cdot b_{t+1}^*(i). \quad (20)$$

The initial condition for  $\psi_t(i)$  is given by:

$$\begin{aligned} \psi_{t_n}(i) &= \text{Prob}[o_1, t'_m, o_2, s_{t_n} = o_2] \\ &= \phi_{t_n}(i), \end{aligned} \quad (21)$$

whose value has been stored at the end of the recursion while calculating  $\phi_t(i)$ .

Finally, the maximum a posteriori (MAP) estimate of user state at time  $t$  is given by:

$$\begin{aligned} s'_t &= \underset{i \in \{1, 2, \dots, N\}}{\text{argmax}} \text{Prob}[s_t = i | o_1, t'_m, o_2] \\ &= \underset{i \in \{1, 2, \dots, N\}}{\text{argmax}} \frac{\text{Prob}[s_t = i, o_1, t'_m, o_2]}{\text{Prob}[o_1, t'_m, o_2]} \\ &= \underset{i \in \{1, 2, \dots, N\}}{\text{argmax}} \psi_t(i), \forall t = t_n - 1, \dots, t_m + 1. \end{aligned} \quad (22)$$

---

**Algorithm 1** Estimating missing states between observations  $O_{t_m} = o_1$  and  $O_{t_n} = o_2$  on a semi-Markov process.

- 1: Input:  $p_{ij}$ ,  $w_a(i)$ ,  $A_i$ , and  $t'_m$  (whose value depends on whether M1, M2, or M3 is used),  $\forall i, j \in \{1, 2, \dots, N\}$
- 2: Output: State estimates  $S' = \{s'_{t_m+1}, \dots, s'_{t_n-1}\}$  and the expected error sequence  $e_s = \{e_{t_m+1}, \dots, e_{t_n-1}\}$
- 3: Initialize  $f_t(i)$  and  $f_t^*(i)$  using equation (8), (9), and (10). Solve by recursion using equation (5) and (7).
- 4: Initialize  $b_t(i)$  and  $b_t^*(i)$  using equation (13), (14), and (15). Solve by recursion using equation (11) and (12).
- 5: Calculate  $\phi_t(i), \forall i \in \{1, \dots, N\}$ , and  $\forall t \in [t_m, t_n]$ , based on equation (17) and (18).
- 6: Calculate  $\psi_t(i), \forall i \in \{1, \dots, N\}$ , and  $\forall t \in [t_m, t_n]$ , based on equation (20) and (21).
- 7: State estimation:

$$s'_t = \underset{i \in \{1, 2, \dots, N\}}{\text{argmax}} \psi_t(i), \forall t = t_n - 1, \dots, t_m + 1.$$

- 8: Expected estimation error ( $\kappa$ : normalization factor):

$$e_t = \kappa \cdot \left\{ 1 - \max_{i \in \{1, 2, \dots, N\}} \psi_t(i) \right\}, \forall t = t_n - 1, \dots, t_m + 1.$$


---

Algorithm 1 estimates missing state information between two neighboring state detections for a discrete-time, semi-Markov process. It can be seen that solving for  $f_t(i)$  and  $b_t(i)$  at time slot  $t$  consumes at most  $A$  multiplications, whereas calculating  $f_t^*(i)$  and  $b_t^*(i)$  requires  $N$  multiplications. Therefore, for an estimation interval with size  $T$ , the computational complexity of the semi-Markov estimation algorithm is

$O(TN(N + A))$ . In comparison, the Markov state estimation mechanism proposed in [33] needs  $T$  matrix multiplications with matrix size  $N \times N$ . This leads to  $O(TN^3)$  computational complexity without any speed-up. Thus, although the complexity of Markov estimation is not affected by the state sojourn duration  $A$ , as the number of states increases, the semi-Markov approach becomes more computationally efficient.

#### D. State estimation performance on simulated processes

In this subsection, we evaluate the performance of Algorithm 1 on two-state, simulated state transition processes. In particular, each state sequence is generated based on pre-specified state sojourn distributions and is observed at a constant frequency, leading to estimation intervals with equal lengths. Within each estimation interval, Algorithm 1 is conducted for state estimation and the estimated sequence is compared to the original sequence to calculate the error.

Recall that one of our goals is to design a real-time state estimation mechanism for sparsely sampled real user state processes. In fact, in many human related traces, state distributions have been found to exhibit heavy tails, e.g., Internet file sizes [7], human contact arrival processes [5], [32], user motion transitions [33], and so on. These types of traces are often modeled as memoryless distributions such as geometric distribution in order to reduce the complexity of study. In this section, we compare the performance of semi-Markov state estimation to state estimation under Markovian assumption, where, no matter what distribution function the state duration follows, the state sequence is always assumed to be a Markov process. The Markovian assumption, although not strictly valid in reality, has been widely used to model user related state traces such as speech [15], [25], mobility [14], [28], and activity [13], [27].

For stationary Markov processes, the state transition probabilities can be estimated using maximum likelihood estimates (MLE) given that annotated data is available [2], [22]:

$$p_{ij} = n_{ij}/n_j, \quad (23)$$

where  $n_{ij}$  is the frequency of state transition from  $i$  to  $j$ , and  $n_j = \sum_i n_{ij}$ .

Estimating state information for missing observations under the Markovian assumption has been discussed in [33]: given that state  $i$  is detected at time  $t_m$ , and state  $j$  is detected at time  $t_n$  ( $t_m, t_n \in T$  and  $t_m < t_n$ ), the most likely state (denoted by  $s'_t$ ) at time  $t$  is selected as

$$s'_t = \underset{k}{\text{argmax}} \left\{ \frac{P_{ik}^{(t-t_m)} \cdot P_{kj}^{(t_n-t)}}{P_{ij}^{(t_n-t_m)}} \right\}, \quad (24)$$

i.e., the MAP state estimation for Markov process does not suffer the sojourn time truncation problem as it implicitly assumes that the underlying state process is memoryless.

We first conduct two sanity checks on semi-Markov estimation by examining its performance on deterministic and geometric distributed state processes. We then select one standard distribution type, namely, binomial distribution, and

examine the performance of semi-Markov estimation. For the widely seen heavy-tailed distributions in real user state traces, we consider Zipf's Law distribution, which is a discrete, power law, and heavy-tailed distribution type.

For each state distribution instance, state estimations are conducted on 10 simulated state processes each containing 1000 state transitions. Specifically, for all non-deterministic processes, Algorithm 1 with variations M1, M2, and M3, as well as Markov estimation mechanism are conducted and their corresponding error ratio  $R$  are compared under different estimation window sizes ranging from 20 to 200.

#### A. Sanity checks: deterministic and geometric distributions

As can be seen from figure 2 (left) that, the semi-Markov estimation with M1 is able to reconstruct the entire sequence without any error under deterministic state distribution, whereas applying Markov estimation leads to undesired performance. This is because when state sojourn durations are constant, knowing how long a state has already spent its sojourn time (M1) will lead to perfect state estimations.

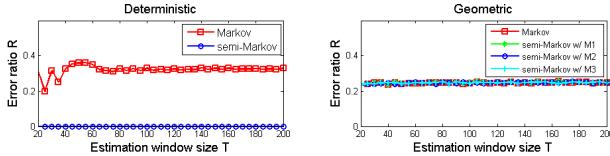


Fig. 2. Algorithm comparisons on deterministic and geometric processes.

Figure 2 (right) shows that semi-Markov and Markov estimation mechanisms lead to very close results when state sojourn times are geometrically distributed<sup>1</sup>. Because Markov process is a special case of semi-Markov process, both estimation mechanisms provide very similar results on a state sequence with geometrically distributed state durations. In fact, different versions of semi-Markov estimation (M1, M2, and M3) provide almost identical results due to the memoryless property of geometric distribution.

#### B. Binomial

It can be seen from figure 3 (left) that the semi-Markov estimation algorithm with M1 outperforms Markov estimation and provides the best performance, whereas M3 leads to the worst performance. Note the similarity between binomial and deterministic distribution, where both pmfs contain peak probability and decay quickly around the concentration. As semi-Markov estimation mechanism provides state estimations strictly based upon state distribution pmf, the “peak-like” feature greatly reduces the estimation error; therefore, knowing history state sequence information (M1) will always lead to better state estimation results.

#### C. Zipf's Law

The results shown in figure 3 (right) illustrate that semi-Markov estimation provides lower estimation error. Table I summarizes the percentage gain provided by different versions of Algorithm 1. It can be seen that while M1 suffers the

<sup>1</sup>As noted in section III-A, although geometric distribution supports infinite set of  $a$  values, we limit it such that  $1 \leq a \leq 100$  in order to satisfy the running condition of Algorithm 1.

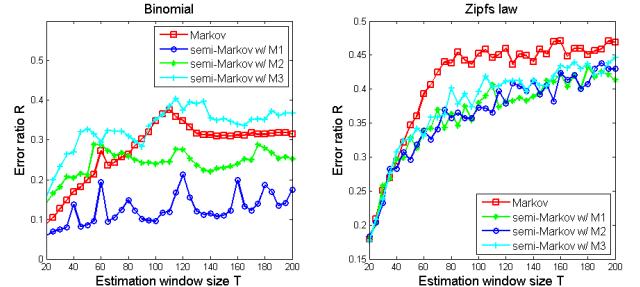


Fig. 3. Algorithm comparisons on binomial and Zipf's law processes.

error propagation problem and M3 assumes unrealistic zero past sojourn time, semi-Markov estimation with M2 leads to the best average performance. Thus, from this point of the paper, we will only consider M2 while evaluating algorithm performance on real data traces that demonstrate heavy-tailed distribution features.

	M1	M2	M3
Avg. Gain	11.09%	11.37%	7.86%
Max Gain	23.79%	19.79%	17.61%

TABLE I  
PERCENTAGE GAIN PROVIDED BY SEMI-MARKOV ESTIMATION OVER MARKOV ESTIMATION (CORRESPONDS TO FIGURE 3 - RIGHT).

#### E. State estimation performance on real traces

To further strengthen our study, the performance of the semi-Markov estimation mechanism is evaluated on real state traces that pertain to smartphone network connectivities (Co-Sphere Project [24]). The traces were collected by researchers at Telematica Instituut in The Netherlands, where twelve participants each carried a smartphone that automatically logged its exposure to cell tower, WiFi access point, and Bluetooth devices for a month. In our study, we pick WiFi and Bluetooth data traces and view the user state as either “Connected” or “Not connected” to infrastructure access points or peer devices, and compare the performance of semi-Markov estimation against Markov estimation.

Note that the stationary assumption will hold on the employed traces, since the data duration (1 month) is much longer as compared to the unit of state duration (10 minutes). In fact, in most applications, although user behavior may vary in different periods, researchers can carefully examine historic data in order to extract durations where different stationary model parameters hold, and state estimation can be conducted.

Similar to equation (23), MLE can be applied in order to estimate the model parameters of a stationary semi-Markov process [37]:

- Transition probabilities estimation:

$$p_{ij} = n_{ij}/n_j, \quad (25)$$

where  $n_{ij}$  is the frequency of state transition from  $i$  to  $j$ , and  $n_j = \sum_i n_{ij}$ .

- Sojourn duration estimation:

$$w_a(i) = n_a(i)/n_i, \forall a \in \{1, 2, \dots, A\}, \quad (26)$$

where  $n_a(i)$  is the number of times that state  $i$  lasts for  $a$  time slots and  $n_i$  is the number of times state  $i$  is visited.

Each user state trace is sampled *periodically* according to the input estimation window size that ranges from 20 to 200. We show the complementary cumulative distribution function (ccdf) of state durations, and demonstrate the estimation error comparison result in figure 4 and 5.

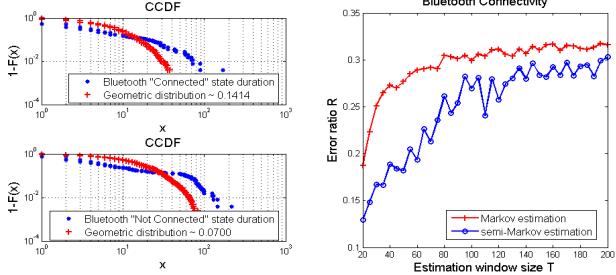


Fig. 4. Bluetooth state distributions and the comparison of estimation error.

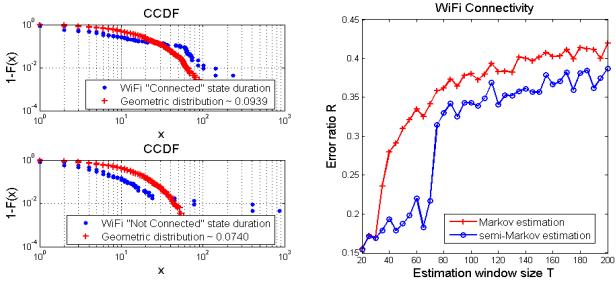


Fig. 5. WiFi state distributions and the comparison of estimation error.

It can be seen that the state distributions of Bluetooth and WiFi connectivity traces exhibit heavier tails than memoryless distributions (the referencing geometric distributions are plotted based on the transition probabilities obtained under Markovian assumption). Clearly, as the semi-Markov process provides more accurate modeling and Algorithm 1 conducts state estimation based on the exact state distribution function, semi-Markov estimation outperforms Markov estimation by achieving at least 14% average gain in terms of accuracy.

#### IV. OPTIMAL SENSOR SCHEDULING WITH ENERGY CONSTRAINT

In practical system designs, a sensor energy consumption budget is often specified in order to ensure a satisfying operation lifetime without recharging the battery. We let  $\xi$  (where  $0 \leq \xi \leq 1$ ) denote the maximum expected energy consumption allowed. Let  $u$  represent a sensor sampling policy that controls the duration the sensor should stay idle under different state detections. We investigate the following constrained optimization problem for a partially observed semi-Markov process: *given an energy budget and state distribution parameters, how should sensor duty cycles be arranged such that the expected state estimation error is minimized?*

We first introduce the formal definition of the two intrinsically conflicting performance metrics, i.e., the expected energy consumption and the expected state estimation error:

**DEFINITION 1:** The expected energy consumption  $E[C]$  of the user state sampling process is defined as

$$E[C] = \lim_{m \rightarrow \infty} \frac{m \cdot 1}{\sum_{k=1}^m I_k}, \quad (27)$$

where  $I_k$  is the length of the  $k$ th estimation interval. Since the energy consumed for each sample is assumed to be 1,  $E[C]$  is therefore equal to the number of estimation intervals divided by the total number of time slots.

**DEFINITION 2:** The expected state estimation error  $E[R]$  is defined as the long term average of per-slot estimation error:

$$E[R] = \lim_{n \rightarrow \infty} \frac{\sum_{t=1}^n e_t}{n} \quad (28)$$

The above constrained optimization problem is formulated as an infinite-horizon CSMDP with the following elements:

- **Decision Epochs  $O$ :** Control decision is made based on the detected user state at each observation slot.

- **System State Space  $X$ :** System state  $x \in X$  is equal to the detected user state at each decision epoch.

- **Action Space  $B$ :** An action  $b \in B$  specifies the duration of idle time under a particular state detection.

- **System Transition Probabilities  $P_{ibj}^s$ :** The semi-Markov transition probability from state  $i$  to state  $j$  in  $b$  time slots.

- **Cost-I  $c_s(y, b)$ :** The expected aggregated semi-Markov estimation error when action  $b$  is taken on state  $y$ .

- **Cost-II  $d_s(y, b)$ :** The intermediate sampling interval when action  $b$  is taken on state  $y$ . Thus  $d_s(y, b) = b$ .

- **Constraint  $\xi$ :** The energy consumption budget.

In this paper, we utilize the LP framework proposed by Feinberg [10], [11] with problem-specific transition probability and cost definitions to solve for the semi-Markov optimal sensing policy  $u_s^*$ .

Note that the "perfect sensing" assumption plays an important role in the above CSMDP formulation, since if observations are erroneous, (say, generated by some emission probabilities), the probability of the underlying true state at each decision epoch will depend on the control policy itself, making it impossible to compute the optimal sensing policy. Although approximations could be introduced by assuming that all observation symbols are emitted under steady state, it is beyond the scope of this paper and we will leave the optimization with erroneous sensing to future work.

#### A. Obtaining $u_s^*$

Let  $\rho(y, a)$  stand for the "occupation measure" of state  $y$  and action  $b$ , i.e., the probability that such state-action pair ever exists in the decision process. The following LP finds the best  $\rho(y, b)$  combinations that meet the energy budget requirement:

$$\text{Minimize } \sum_{y \in X} \sum_{b \in B} \rho(y, b) c_s(y, b) \quad (29)$$

**subject to:**

$$\sum_{y \in X} \sum_{b \in B} \rho(y, b) (\delta_x(y) - P_{ybx}^s) = 0, \forall x \in X, \quad (30)$$

$$\sum_{y \in X} \sum_{b \in B} \rho(y, b) = 1, \quad (31)$$

$$\rho(y, b) \geq 0, \forall y, b, \text{ and} \quad (32)$$

$$\sum_{y \in X} \sum_{b \in B} \rho(y, b) (1 - b \cdot \xi) \leq 0, \quad (33)$$

The constraint given in (30) describes that the outgoing and incoming rate of any state need to be the same. The constraints (31) and (32) define  $\rho(y, b)$  as a probability measure. The inequality constraint given in (33) guarantees that the expected energy usage is less than the energy constraint value  $\xi$ , i.e.:

$$\frac{\sum_y \sum_b \rho(y, b)}{\sum_y \sum_b \rho(y, b) \cdot b} \leq \xi, \quad (34)$$

which leads to the energy constraint in (33).

The above LP is an enhancement to the standard form proposed in [10], where, instead of obtaining a policy that does not consider history, the previous sojourn time is taken into account by defining the system state transition probability  $P_{ybx}^s$  and the intermediate estimation error  $c_s(y, b)$  based on the expected past sojourn duration (defined in M2). In particular, given the estimation interval size  $b$ , starting state  $y$ , and ending state  $x$ , the interval transition probability  $P_{ybx}^s$  could be calculated at step 5 of Algorithm 1, i.e.:

$$P_{ybx}^s = \phi_{t_n}(x), \quad (35)$$

under appropriate input conditions:

$$o_1 = y, o_2 = x, t_n - t_m = b, \text{ and} \quad (36)$$

$$t_m - t'_m = \sum_{a=1}^A \sum_{\tau=1}^a \frac{w_y(a) \cdot \tau}{a} - 1. \quad (37)$$

Similarly, at step 8 of Algorithm 1, the aggregated estimation error of that estimation interval can be expressed as:

$$e_{sum} = \sum_{t=t_m}^{t_n} e_t. \quad (38)$$

Let  $e_{sum}^{ybx}$  denote the aggregated estimation error under condition (36), the intermediate cost  $c_s(y, b)$  could then be expressed as the following:

$$c_s(y, b) = \sum_x P_{ybx}^s \cdot e_{sum}^{ybx}. \quad (39)$$

Once the linear programming is solved for optimal occupation measures  $\rho^*(y, b)$ , the semi-Markov optimal policy  $u_s^*$  can be constructed such that the probability of taking action  $b$  at state  $y$  is equal to  $\frac{\rho^*(y, b)}{\rho_y^*}$ , where  $\rho_y^* = \sum_{b \in B} \rho^*(y, b), \forall y \in X$ .

#### B. Performance of $u_s^*$ on real user state traces

The performance of  $u_s^*$  is evaluated on the same set of real user state traces introduced in section III-E. As  $u_m^*$  is optimal for Markov processes, we explore whether  $u_s^*$  could outperform  $u_m^*$  by achieving a better trade-off between energy consumption and state estimation error, on user state traces that are not strictly Markovian. In particular, the following steps are executed for each user state trace:

- (1) Estimate model parameters using (25) and (26).
- (2) Compute  $u_s^*$  using LP (29) - (33).
- (3) Compute  $u_m^*$  according to techniques proposed in [33].
- (4) Apply both policies on the data trace and conduct semi-Markov and Markov estimation correspondingly, and obtain the estimation error as well as the energy usage.

Since the optimization framework is stochastic and is constrained on the expected energy usage, the true energy consumption when executing the policy on a particular trace may not exactly reflect the given energy budget. We thus show the results of estimation error vs. the actual energy usage in figure 6. To better visualize the performance gain, the linear fit of the energy-error pairs under different policies are shown in figure 6. It can be seen that  $u_s^*$  yields a better trade-off between energy consumption and state estimation error as compared to  $u_m^*$ , by producing 12.04% and 14.19% less estimation error on average while using the same energy consumption in each respective case.

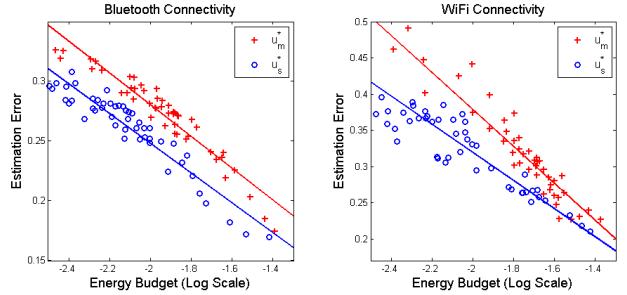


Fig. 6. Performance of  $u_m^*$  vs.  $u_s^*$  CoSphere traces.

Note that the comparison under large energy budgets is not shown in figure 6 because  $u_s^*$  and  $u_m^*$  are found to provide comparably low estimation errors when energy budget is not stringent. In fact, this can be easily concluded from figure 3 which suggests that the estimation errors are indeed similar under small estimation window sizes. However, real mobile sensing systems often operate under small energy budgets and deal with sparse observations, in which case the semi-Markov estimation as well as the semi-Markov optimal policy lead to the best improvement and are therefore desirable implementation choices.

## V. POLICY IMPLEMENTATION AND EVALUATION IN A REAL MOBILE SENSING SYSTEM

In traditional designs of mobile sensing systems such as [21], [34], there is a lack of theoretical optimization framework that targets at achieving the best trade-off between state detection/estimation accuracy and energy consumption. In this section, we aim to introduce such a mechanism by implementing the semi-Markov optimal policy  $u_s^*$  on a basic human activity recognition system, and providing benchmark performance evaluation of  $u_s^*$ ,  $u_m^*$ , and uniform sampling.

#### A. Design and implementation of the evaluation system

The designed mobile sensing system adopts a fully automated, client-server based architecture whose function flow is shown in figure 7. Note that the automated feature minimizes the burden of user-device interaction since zero user input is required. As figure 7 illustrates, first, mobile client executes the *Behavior Learning Module*, which collects historic user state information and calculates the state distribution parameters from recorded data. This procedure allows specific optimal control parameters to be derived for users with different habits. The state distribution parameters will be sent to the *Optimal*

*Policy Calculation Module* on the back-end server in order to compute the user-specific  $u_s^*$  (and  $u_m^*$ ). The optimal sensor duty cycles are then transferred back to mobile client to begin energy efficient sensing, which is realized via the *Energy Efficient Sensing Module*. Finally, the partially observed data sequence will be transferred to *State Estimation Module* which is responsible of estimating missing state information.

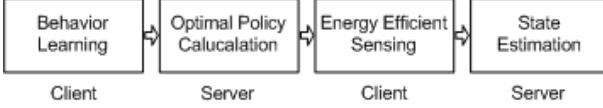


Fig. 7. Operation flow of the fully automated evaluation system.

Note that currently, the optimal policy calculation and state estimations are conducted on back-end server due to its higher computing power and the availability of mathematical softwares/libraries. We implement a prototype of the system architecture shown in figure 7 as a basic activity recognition system on Nokia N95 devices (mobile client) and desktop PCs (back-end server) using Symbian C++ and Matlab, which communicate to each other via socket connections.

The mobile client is designed to differentiate two human activity states (“Stable” and “Moving”) based on accelerometer sensing. In particular, the acceleration values on x, y, and z axis of the accelerometer is continuously read for a certain duration, and the standard deviation of the magnitudes of all three axes readings is calculated. The user activity is classified based on whether the standard deviation is below or above some given threshold. Recall that our goal is not to develop sophisticated activity classification algorithms, but to explore how to derive and implement the stochastic optimal policy on the mobile client and whether it is able to provide satisfying performance in real system implementations. In fact, in [34] we showed that a single accelerometer is able to differentiate “Moving” and “Stable” with negligible error.

In our evaluation system, the user behavior learning phase is conducted for seven consecutive days, during which, the user activity trace is fully detected and logged in the mobile device file system. The symbol “0” and “1” are used to represent different user states so that the state distribution parameters can be easily calculated using equation (25) and (26). During the learning phase, the mobile device is frequently charged to maintain continuous sensing requirement.

On the server side, the policy calculation, state estimations, as well as socket connection setup are all accomplished in Matlab which provides standard LP solvers. Upon receiving the state distribution parameters (in string format), the server calculates the semi-Markov optimal sensing policy  $u_s^*$  and transfers it back to the client in terms of the average idle interval size under different state detections, such that at each decision epoch (the length of each time slot is 10 seconds), the client will determine how long to turn off the sensor until the next observation.

#### B. Performance evaluation

In this paper, we only focus on providing benchmark performance measurement of the mobile client while implementing

different sensing policies, and we leave the potential optimization on the trade-off between communication and computation cost and the investigation of advanced learning techniques such as reinforcement learning to future study.

Five independent experiments have been conducted and their setups are introduced below. During each experiment, the empirical devices are carried by a participant as his personal cell phone but with no usage other than activity sensing.

**Exp-1** ( $\xi = 1$ ): the accelerometer is continually sampled and activity is classified in each time slot. This sets the benchmark of the battery lifetime when sensor is fully activated.

**Exp-2** ( $\xi = 0$ ): this experiment provides a bottom-line comparison and measures how long the same empirical device lasts without any sensing task.

**Exp-3,4,5** ( $\xi = 0.05$ ): the accelerometer on the main empirical smartphone is operated according to the sensing intervals specified by  $u_s^*$ ,  $u_m^*$ , and uniform sampling, respectively. A second smartphone is used throughout these experiments which samples its accelerometer in every time slot to provide ground truth user state record. This second device is frequently charged to ensure long operating duration.

To evaluate the effectiveness of the policies, we measure the device battery lifetime from the time the smartphone receives the optimal duty cycle instructions (when the battery is still fully charged) until the battery is completely drained while executing sensing policies accordingly. The partially sampled user state sequences are reconstructed off-line, and are compared to the ground truth in order to obtain the estimation error. The empirical results are shown in table II.

	Policy	Battery Life	Estimation Error (%)
exp-1	Full	52 hours	0
exp-2	None	236 hours	$\infty$
exp-3	$u_s^*$	160 hours	9.96%
exp-4	$u_m^*$	158 hours	13.8%
exp-5	Uniform	150 hours	19.38%

TABLE II  
SMARTPHONE BATTERY LIFETIME AND STATE ESTIMATION ERROR.

It can be seen from table II that implementing  $u_s^*$  achieves the longest device battery lifetime among all three energy-saving policies and provides 27.8% and 48.6% gain on estimation accuracy over  $u_m^*$  and uniform sampling respectively.

To better visualize policy comparison, we reapply all policies on a subset of data collected during the experiment, and show the distribution of sample points as well as the growth of corresponding estimation error in figure 8. Note that although figure 8 displays the motion trace (Stable vs. Moving) within approximately 1 hour where the user is fairly active, the actual empirical duration contains many significant idle periods (e.g., night hours). The optimization framework proposed in this paper is able to provide better estimation performance by finding  $u_s^*$ , which samples more aggressively when the user is found active, and increases the interval size when the user is in idle period in order to balance the overall energy consumption.

## VI. CONCLUSION AND FUTURE WORKS

Modeling user state traces using Markov model is often unrealistic and could lead to undesired state estimation performance. In this paper, we use a more general class of processes

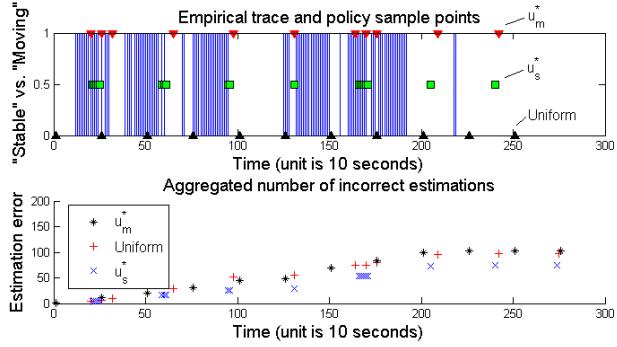


Fig. 8. Data trace (solid line indicates Moving and blank space means Stable), policy sample points, and aggregated state estimation error for this interval.

- semi-Markov process - to model user state traces, and propose a generic semi-Markov state estimation mechanism that can be executed in real-time in order to estimate the most likely user state while sensor observations are missing. We also obtain the semi-Markov optimal sensing policy by solving the CSMDP problem for intermittently sampled semi-Markov processes, which is able to minimize the expected state estimation error while maintaining an energy consumption budget. Finally, we provide benchmark performance evaluation of three energy-saving policies on a novel client-server based, fully automated human activity recognition system and show that semi-Markov optimal policy is the desirable choice as compared to Markov-optimal policy and uniform sampling.

For future research directions, we plan to investigate a more comprehensive theoretical optimization framework where (1) multiple sources of energy consumption such as computation and communication are all taken into consideration, (2) state detection could be erroneous, such that sensor selection is available based on their different energy consumption and detection accuracy and (3) different importance levels are associated with different states.

## REFERENCES

- [1] P. Aghera, D. Fang, T. Simunic Rosing, and K. Patrick, *Energy management in wireless healthcare systems*, IPSN, 2009.
- [2] M. S. Barlett, *The frequency goodness of fit test for probability chains*, Mathematical Proceedings of the Cambridge Philosophical Society, 1951.
- [3] L. Benini, A. Bogliolo, G. A. Paleologo, and G. D. Micheli, *Markov decision processes for control of a sensor network-based health monitoring system*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 18, 1998, pp. 813–833.
- [4] O. Cakmakci, J. Coutaz, K. V. Laerhoven, and H. werner Gellersen, *Context awareness in systems with limited resources*, AIMS, 2002.
- [5] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, *Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding*, Technical Report 617, University of Cambridge, 2005.
- [6] T. L. Cheung, K. Okamoto, F. Maker, X. Liu, and V. Akella, *Markov decision process (mdp) framework for optimizing software on mobile phones*, EMSOFT, 2009.
- [7] Mark E. Crovella and Azer Bestavros, *Self-similarity in world wide web traffic evidence and possible causes*, IEEE/ACM Transactions on Networking 5 (1996), 835–846.
- [8] E. Cuervoy, A. Balasubramanian, D. Cho, A. Wolmanx, S. Saroiu, R. Chandrax, and P. Bahl, *Maui: Making smartphones last longer with code offload*, MobiSys, 2010.
- [9] F. R. Dogar, P. Steenkiste, and K. Papagiannaki, *Catnap: Exploiting high bandwidth wireless interfaces to save energy for mobile devices*, MobiSys, 2010.
- [10] E. A. Feinberg, *Constrained semi-markov decision processes with average rewards*, Mathematical Methods of Operations Research, vol. 39, 1994, pp. 257–288.
- [11] E. A. Feinberg and A. Shwartz, *Constrained markov decision models with weighted discounted criteria*, Mathematical Methods of Operations Research 20 (1995), 302–320.
- [12] J. D. Ferguson, *Variable duration models for speech*, Symposium on the Application of Hidden Markov Models to Text and Speech, 1980.
- [13] J. Gao, E. G. Hauptmann, A. Bharucha, and H. D. Wactlar, *Dining activity analysis using a hidden markov model*, ICPR, 2004.
- [14] M. Grossglauser and D. Tse, *Mobility increases the capacity of ad-hoc wireless networks*, vol. 10, 2002, pp. 477–486.
- [15] J. Jaffe, L. Cassotta, and S. Feldstein, *Markovian model of time patterns of speech*, Science, Volume 144, 1964.
- [16] R. Jurdak, P. Corke, D. Dharman, and G. Salagnac, *Adaptive gps duty cycling and radio ranging for energy-efficient localization*, SenSys, 2010.
- [17] S. Kang, J. Lee, H. Jang, H. Lee, Y. Lee, S. Park, T. Park, and J. Song, *Seemon: scalable and energy-efficient context monitoring framework for sensor-rich mobile environments*, MobiSys, 2008.
- [18] D. H. Kim, Y. Kim, D. Estrin, and M. B. Srivastava, *Sensloc: Sensing everyday places and paths using less energy*, SenSys, 2010.
- [19] A. Krause, M. Ihmig, E. Rankin, S. Gupta, D. Leong, D. P. Siewiorek, A. Smailagic, M. Deisher, and U. Sengupta, *Trading off prediction accuracy and power consumption for context-aware wearable computing*, IEEE International Symposium on Wearable Computers, 2005.
- [20] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, *Energy-accuracy aware localization for mobile devices*, MobiSys, 2010.
- [21] E. Miluzzo, N. Lane, K. Fodor, R. Peterson, S. Eisenman, H. Lu, M. Musolesi, X. Zheng, and A. Campbell, *Sensing meets mobile social networks: The design, implementation and evaluation of the cenceme application*, SenSys, 2008.
- [22] K. Murphy, *Dynamic bayesian networks: Representation, inference and learning*, Ph.D. dissertation, University of California, Berkeley, 2002.
- [23] J. Paek, J. Kim, and R. Govindan, *Energy-efficient rate-adaptive gps-based positioning for smartphones*, MobiSys, 2010.
- [24] A. Peddemors, H. Eertink, and I. Niemegeers, *Density estimation for out-of-range events on personal mobile devices*, MobilityModel, 2008.
- [25] L. R. Rabiner, *A tutorial on hidden markov models and selected applications in speech recognition*, Proceedings of the IEEE, 1989.
- [26] E. Rozner, V. Navda, R. Ramjee, and S. Rayanchu, *Napman: Network-assisted power management for wifi devices*, MobiSys, 2010.
- [27] D. Sanchez, M. Tentori, and J. Favela, *Hidden markov models for activity recognition in ambient intelligence environments*, ENC, 2007.
- [28] G. Sharma and R. R. Mazumdar, *Scaling laws for capacity and delay in wireless ad hoc networks with random mobility*, ICC, 2004.
- [29] G. Thatte, M. Li, A. Emken, U. Mitra, S. Narayanan, M. Annavaram, and D. Spruijt-Metz, *Energy-efficient multihypothesis activity-detection for health-monitoring applications*, EMBC, 2009.
- [30] G. Thatte, V. Rozgic, M. Li, S. Ghosh, U. Mitra, S. Narayanan, M. Annavaram, and D. Spruijt-Metz, *Optimal allocation of time-resources for multihypothesis activity-level detection*, DCOSS, 2009.
- [31] A. Thiagarajan, L. S. Ravindranath, K. LaCurts, S. Toledo, J. Eriksson, S. Madden, and H. Balakrishnan, *Vtrack: Accurate, energy-aware traffic delay estimation using mobile phones*, SenSys, 2009.
- [32] W. Wang, V. Srinivasan, and M. Motani, *Adaptive contact probing mechanisms for delay tolerant applications*, MobiCom, 2007.
- [33] Y. Wang, B. Krishnamachari, Q. Zhao, and M. Annavaram, *Markov-optimal sensing policy for user state estimation in mobile devices*, IPSN, 2010.
- [34] Y. Wang, J. Lin, M. Annavaram, Q. A. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh, *A framework of energy efficient mobile sensing for automatic user state recognition*, MobiSys, 2009.
- [35] W. Wu, M. Batalin, L. Au, A. Bui, and W. Kaiser, *Context-aware sensing of physiological signals*, Proceedings of EMBS, 2007.
- [36] W. Wu, A. Bui, M. Batalin, D. Liu, and W. Kaiser, *Incremental diagnosis method for intelligent wearable sensor systems*, IEEE Transactions on Information Technology in Biomedicine, vol. 11, 2007, pp. 553–562.
- [37] S. Yu, *Hidden semi-markov models*, Artificial Intelligence, 2009.
- [38] S. Yu and H. Kobayashi, *A hidden semi-markov model with missing data and multiple observation sequences for mobility tracking*, Signal Processing, Volume 83, Issue 2, 2003.
- [39] Z. Zhuang, K. Kim, and J. P. Singh, *Improving energy efficiency of location sensing on smartphones*, MobiSys, 2010.