Computers and Electrical Engineering 58 (2017) 310–326

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

DMP-IOT: A distributed movement prediction scheme for IOT health-care applications[☆]

Azadeh Zamanifar[*], Eslam Nazemi, Mojtaba Vahidi-Asl

Computer Science and Engineering Faculty, Shahid Beheshti University, Tehran, Iran

A R T I C L E I N F O

Article history:
Received 15 May 2016
Revised 14 September 2016
Accepted 14 September 2016
Available online 22 September 2016

Keywords:
Mobile IP-based WSN
Movement prediction
Hidden Markov model
Mobility management
Health-care

A B S T R A C T

Mobility prediction in IP-based WSNs makes it possible to predict the next movement direction of mobile sensor nodes, which results in less power consumption and delay during handoff. The previous direction detection approaches need specific hardware facilities and impose considerable overhead during handoff. The advantage of DMP-IOT is distributing the learning model data around static sensors of the proposed tree, after the training phase. DMP-IOT includes a recovery mechanism that avoids disconnection of the mobile sensor node(s) to the network in case of a false prediction. The simulation results show about 25% improvement of DMP-IOT in saving power consumption and reducing handoff delay and packet loss, compared to movement direction approaches in similar works. The accuracy of the proposed movement prediction scheme is 83%, in average, which is validated by $t - Student$ statistical test. Comparing the second-order Hidden Markov Model (HMM) with ANN reveals the superiority of the second-order HMM model in our application.

1. Introduction

The number of advanced digital devices has been dramatically increased in recent years. Due to their powerful hardware and software facilities, sophisticated platforms have been provided for connecting all those devices via the Internet like structure [1], known as Internet of Things (IOT). The mobile IP-based sensor network is a popular and well-known infrastructure of IOT where each mobile sensor node can have an IPv6 address. The Wireless Sensor Networks (WSNs) have been widely used in health-care and military applications where the sensor nodes are worn on or implanted into the human body to measure the vital parameters [2]. For Body-to-Body ubiquitous health-care applications, accurate mobility prediction is necessary to perform critical tasks related to medical data routing among mobile WBANs. The critical tasks include call admission control, congestion control, the reservation of network resources, pre-configuration of services and QoS provisioning. The mobile WSNs are flexible environments with dynamic changes in which the number of sensors fluctuates in each network [3]. In poor infrastructure environments, if the mobile sensor nodes directly communicate to the gateway, a considerable amount of mobile sensor node's power would be consumed. Thus, for reducing the power consumption of mobile sensor nodes, a mobility management procedure is used with the aid of static sensors which have fewer power constraints compared to the mobile sensor nodes [4].

[☆] Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. M. Malek.
[*] Corresponding author.
E-mail addresses: a_zamanifar@sbu.ac.ir (A. Zamanifar), nazemi@sbu.ac.ir (E. Nazemi), mo_vahidi@sbu.ac.ir (M. Vahidi-Asl).

http://dx.doi.org/10.1016/j.compeleceng.2016.09.015
0045-7906/© 2016 Elsevier Ltd. All rights reserved.

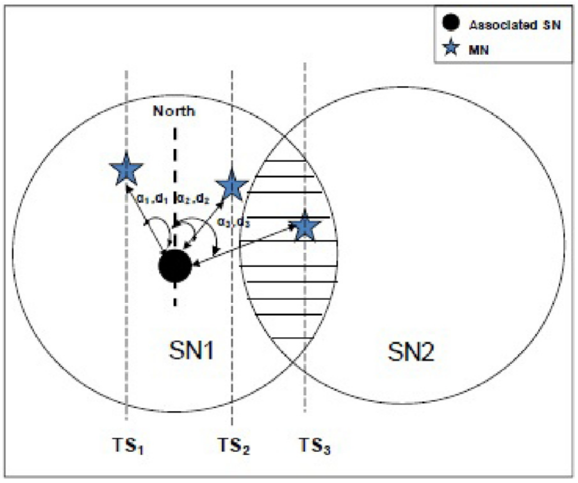

Fig. 1. Movement direction detection by AOA method [9].

In IP-based WSNs, a mobility management procedure typically has four main phases: (1) detection of movement (2) identifying the direction of motion (3) buffering data destined to the mobile sensor node during the handoff (4) sending the new location of the mobile sensor node to the gateway. The first two phases of the procedure have a great impact on handoff cost because they consume a significant amount of power. Furthermore, if identifying the direction of movement is done by mistake, the mobile sensor node's connection to the network is broken which has a great negative impact on safety critical applications like health-care.

The typical WSN is designed to be efficient in energy consumption by reducing the number of message exchanges while mobile sensor node(s) changes its point of attachment to the network. Another primary concern of WSNs is reducing packet loss caused by handoff delay. An efficient movement direction detection approach in WSNs should consider both of these factors. The previous works mostly detect the direction of a movement by using AOA (Angle of Arrival) method with the aid of additional hardware like directional antennas, antenna arrays, etc. [5]. AOA is defined as the angle between the propagation direction of an incident wave and some reference direction, which is known as orientation. Orientation, defined as a fixed direction against which the AOAs are measured, is represented as degrees in a clockwise direction from the North. When the orientation is 0 or pointing to the North, the AOA is absolute. Otherwise, it would be relative. One standard approach to obtaining AOA measurement is to use an antenna array on each sensor node [6–8]. As shown in Fig. 1, 'the star' shows mobile sensor node, abbreviated as MN, position at timestamps TS_1, TS_2, and TS_3, respectively. The associated Static Node abbreviated as SN, measures AOA from a signal that it receives from the MN. Moreover, the associated SN also estimates the distance between the MN and itself by using Received Signal Strength (RSS) of the packets. As shown in Fig. 1, let α_1, d_1 ; α_2, d_2 ; α_3, d_3 be the angles (in degrees) and the distances between the SN1 and the MN, at timestamps TS_1, TS_2 and TS_3, respectively. Considering the angle and distance information, the SN can get the location coordinates MN. The accuracy of this method has not been evaluated before. Furthermore, this method needs specific hardware that is not always available in real applications considering that implementation of networks with this feature is difficult. Moreover, if the directional antenna malfunctions or fails, detecting the direction of movement would be impossible. Another important issue is that if a patient's movement direction suddenly changes, the direction predicted by the AOA method would be incorrect, which causes dis-connectivity of the mobile sensor node from the network.

The direction of movements can also be detected by broadcasting a message and measuring RSS of messages exchanged between mobile and neighboring static nodes [10–13]. Some other works use LQI (Link Quality Indicator) and RSSI (Received Signal Strength Indicator) together to predict movement directions, more accurately. The RSS method, which is the easiest one, requires a considerable number of message exchanges, which decreases the lifetime of mobile sensor nodes. Similar to AOA, whenever the direction of a mobile sensor node suddenly changes, it fails to find the correct movement direction.

To address the mentioned issues, we have designed a novel network and Distributed Movement Prediction scheme, DMP-IOT, which predicts the movement direction of mobile sensor nodes in health-care applications to reduce the handoff cost of the mobile sensor node(s). To this end, second-order Hidden Markov Model has been customized for mobility prediction in the health-care application. DMP-IOT includes a false prediction recovery mechanism which avoids disconnection of the mobile sensor node(s) to the network in case of incorrect prediction of patient's movement. With the recovery mechanism, handoff is not affected by false prediction and the connectivity of the mobile sensor node to the network is not disrupted. It is worth noting that vehicular network in which power consumption and continuous connectivity are not their issues is out of the scope of this paper, accordingly. To our knowledge, no learning method has been applied to modeling movement in mobile IP-based WSNs.

The rest of the paper is organized as follows. In Section 2, we discuss the related works on movement prediction. The proposed approach is described in Section 3. The simulation results are presented in Section 4. We conclude the paper with a summary in Section 5.

2. Background and related work

Since, there is no movement prediction method for IP-based mobile sensor networks, we review the movement direction prediction methods in similar contexts.

Movement prediction can be done either for indoor or outdoor location tracking [14,15]. For outdoor location tracking, predicting the pedestrian trajectory in an outdoor area is considered while in indoor tracking, a person's next location inside a building is taken into account. The outdoor tracking data are analyzed to find frequent and popular routes, path clustering or finding abnormal paths.

In this paper, for health-care applications, we focus on indoor location tracking of patients. The mobility model could be either (1) synthetic that is extracted from a random walk , (2) simple probability model or (3) trace-based mobility model that is extracted from synthetic moving data [16]. Movement prediction can be provided by applying different algorithms based on different features. Markov, Hidden Markov, Artificial Neural Network (ANN) and Bayesian network techniques are mainly applied for prediction in the literature [17–19].

Baratchi et al. have proposed a model for mobile data movement based on HMM [20]. They have also considered the duration of each state by designing a hierarchical model with different state granularity. The super states composed of small states contain behaviors which are repeated, periodically. However, this solution is appropriate for outdoor tracking. We have decreased the complexity of HMM by distributing the model with the aid of our proposed tree scheme where each tree node maintains a part of the model that it needs for movement prediction.

Gellert et al. [19] use HMM for movement prediction in an office. They have evaluated their approach with a different number of hidden states and also with HMM series 1–3 and Neural Network. However, they predict the next movement location of a person if the source location is not his own room which makes the prediction very simple. Because of this assumption, the accuracy of their proposed approach is high. Tran et.al [21] have proposed an approach based on decision tree to predict the next location of a mobile user. Their decision tree can detect movements to new places on different days of the week. They have considered properties for the time of the day, weekday, place ID, start and leave time for each place. However, a decision tree is a slow solution for an online prediction and cannot model the hidden states such as the thoughts of moving objects. It only considers the current location of a user and it is not a probabilistic model.

Furey et al. [22] have formulated the movement prediction problem as a graph in which the most frequently visited place is represented as a node. The paths between areas are considered as the graph edges. They use a discrete Bayes filter to model actions and movements. However, it is a centralized system in which every decision should be made on the server, and the real-time calculation of movement probabilities makes it a slow approach. It also assumes that movement probability is independent of the previous states which seems not to be realistic.

Duong et al. use a log file of node mobility history in mobile IP networks to predict next movement of mobile sensor nodes. They generate the transactional database from the log file of node mobility history, discover all frequent movement patterns in the database and generate all mobility rules using the regular mobility patterns. However, it needs a significant amount of data in learning phase which is not adequate for real time and power constrained mobile sensor networks [23].

In [24] the authors show the suitability of an ANN, implemented in ad-hoc routing protocol, for predicting human mobility in opportunistic network scenarios as well as showing the effects of implementing a mobility model on reliability, delay and delay variation. However, as the model update process is done inside the nodes, it increases power consumption that is an issue for the sensors.

The majority of the movement direction prediction methods that are applied in different contexts, fail to consider the temporal variable in their model, completely [20,25]. It means that they do not consider that the pattern of movements may vary at different times of the day. For example, if a patient is in the kitchen, to determine the next movement of the patient, it is important to know where he stayed before going to the kitchen which implicitly represents the time of the day. Another drawback is that they do not propose a scheme consistent with their proposed method. Our proposed approach takes advantage of tree structure where each node maintains a part of the model related to the probability of movement from/to itself to/from the neighboring cells.

3. The proposed approach

In IP-based mobile WSN, which is designed for health-care applications, we assume that there are n mobile sensor nodes and m static sensors. The static nodes handle the movement overhead of mobile sensor nodes, none of the nodes is aware of its location. Every static sensor is assumed to be in the range of its neighbors. Each mobile sensor node sends the data through the nearest static node, so-called candidate node, to the gateway. A candidate node is one of the leaves of the tree which is placed in the middle of a cell where the mobile sensor node is located, at the moment. The goal is to maintain the connectivity of mobile sensor node(s) to the gateway during handoff.

DMP-IOT has three main parts as shown in Fig. 2 including the initial setup, movement prediction and movement management that are described in the following sub-sections. For more convenience, the serial number in each box represents

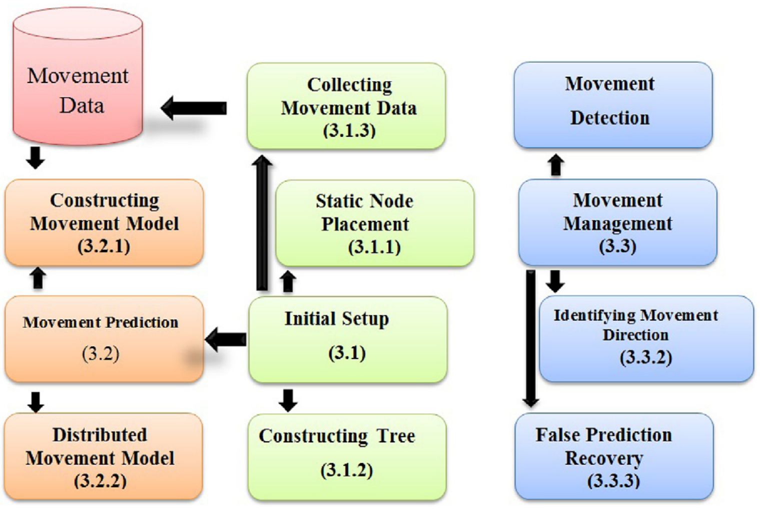
Movement Data
Collecting Movement Data (3.1.3)
Movement Detection
Constructing Movement Model (3.2.1)
Static Node Placement (3.1.1)
Movement Management (3.3)
Movement Prediction (3.2)
Initial Setup (3.1)
Identifying Movement Direction (3.3.2)
Distributed Movement Model (3.2.2)
Constructing Tree (3.1.2)
False Prediction Recovery (3.3.3)

Fig. 2. DMP-IOT architecture.

the number of sub-section which describes the corresponding module. Initial setup of DMP-IOT is described in 3.1, where the static node placement, DMP-Tree construction and movement data collection are carried out. Subsequently, in Section 3.2, the movement prediction model is generated and distributed using the constructed tree. In Section 3.3, the movement management module is described. The overall steps of the proposed movement prediction approach in our suggested network scheme is shown in Fig. 3. As shown in the figure, during the training phase, the patient's tracking data is forwarded to the gateway via the DMP-Tree where the HMM model, as a table, is constructed. Then it is divided into sub-tables, each of them is carrying data required for predicting the direction of the patient's movement considering each cell of the area. These sub-tables are forwarded through DMP-Tree and distributed over static leaf nodes, positioned at the center of each cell.

3.1. Initial setup

At the initial network setup, firstly we put static nodes DSHMP-Tree, detailed in Section 3.1.1. Then, build the tree in Section 3.1.2. Subsequently, collecting movement data is described in Section 3.1.3.

3.1.1. Static node placement

The algorithm for static node placement of DMP-Tree is depicted in Table 1 where *SensorCount* is the total number of static nodes, *tree* is the variable which holds the tree information, *Area* is the monitoring area and *pos* holds the two dimensional positions of static nodes. The monitoring area is assumed to be a Cartesian plane with coordinates X and Y where the origin is the left down corner of the area. Parameter k holds the current tree level and L holds the cell's length, which could be in square meters. At first, the static nodes as the leaves of the tree are placed at the middle of cells (line 4). For two given leaves, the parent is a static node which is placed at a point where its x location is the middle of its children's x values and its y value is the y value of the children plus $L/2$. Similarly, the static nodes at higher levels are positioned (line 7–12). For a given node z, the id of its left child is twice as the ID of z and the id of its right child is twice as the ID of z plus one. After the placement of nodes, each node finds its parent and children. Firstly, the root sends the ID of its two children. Subsequently, each child of the root that receives its ID, sends the ID of its children, in a similar way. This is done recursively until the leaves of the tree set their IDs.

3.1.2. Tree construction

For efficient distribution of the prediction model across the monitoring area, we need an efficient tree structure. The proposed tree, DMP-Tree, also simplifies addressing, routing and collecting data for model construction. The most important advantage of DMP-Tree is facilitating the process of predicting the next movement of a patient, stayed in a cell, by communication within the neighboring cells. In the proposed tree, each two leaves in two corresponding neighboring cells have a common node that is in the range of both adjacent cells. Firstly, to make the network scalable, we divide the monitoring area into equal sized cells. In other words, the home area is considered as a set of the cells with length L. For the area shown in Fig. 4, the transmission range of the DMP-Tree's leaf nodes are computed as Eq. (1).

$$R = \sqrt{2}L \tag{1}$$

For example, in Fig. 4 the parent of two leaf nodes *1, 2* is node (1, 2), located at the common up corner of corresponding cells, and the parent of two non-leaf nodes (1, 2) and (3, 4) is *A*, located in the middle of the line connecting the children.

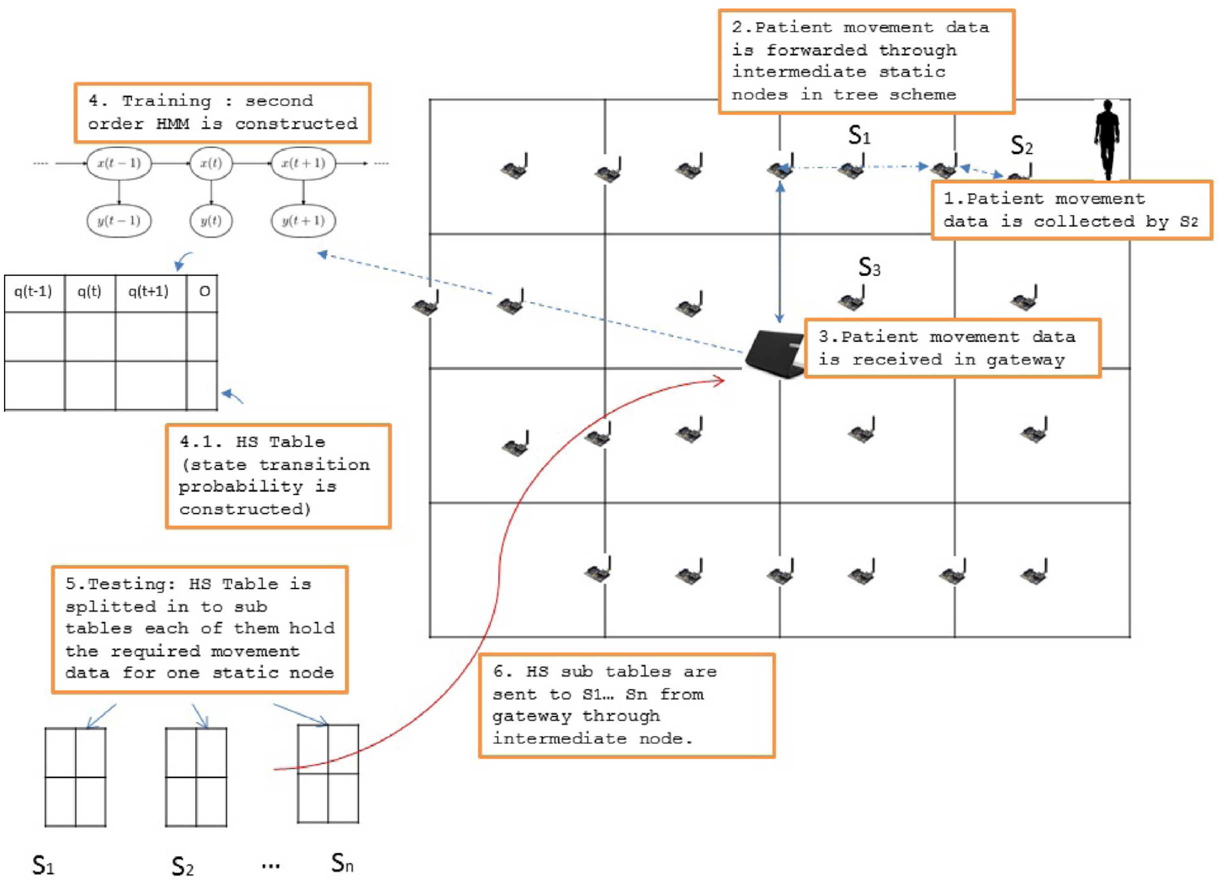
2.Patient movement data is forwarded through intermediate static nodes in tree scheme
4. Training : second order HMM is constructed
S1
S2
1.Patient movement data is collected by S2
S3
q(t-1)
q(t)
q(t+1)
O
3.Patient movement data is received in gateway
4.1. HS Table (state transition probability is constructed)
5.Testing: HS Table is splitted in to sub tables each of them hold the required movement data for one static node
6. HS sub tables are sent to S1… Sn from gateway through intermediate node.
S1
S2
...
Sn

Fig. 3. The proposed DMP-IOT movement prediction steps+ network scheme.

Table 1
Algorithm for tree construction.

```
Tree Construction Algorithm
TreeConstruction(Area, SensorCoverage, pos)
{
1.  $SensorCount = 4 * Area/(sensorCoverage^2)$
2.  tree= complete binary trees calculated from ($SensorCount = 2^x$)
3.  i=tree.LeavesCount
4.  put Nodes[0..i] in pos (0..i)
5.  k=1
6.  While $k \neq 2 * i - 1$
7.      put Nodes[k/2+i] in pos(k/2+i)
8.      If k $\neq$ 1 then
9.          pos(k/2+i)=(pos(k)+pos(k-1))/2
10.     else
11.         pos(k/2+i]=(pos(k)+pos(k-1))/2+L/2
12.     EndIf
14.     parent(Nodes[k-1])=Nodes[k/2+i]
15.     parent(Nodes[k])=Nodes[k/2+i]
16.     add Nodes[k-1] to child(Nodes[k/2+i])
17.     add Nodes[k] to child(Nodes[k/2+i])
18.     k=k+2
19. Endloop
}
```

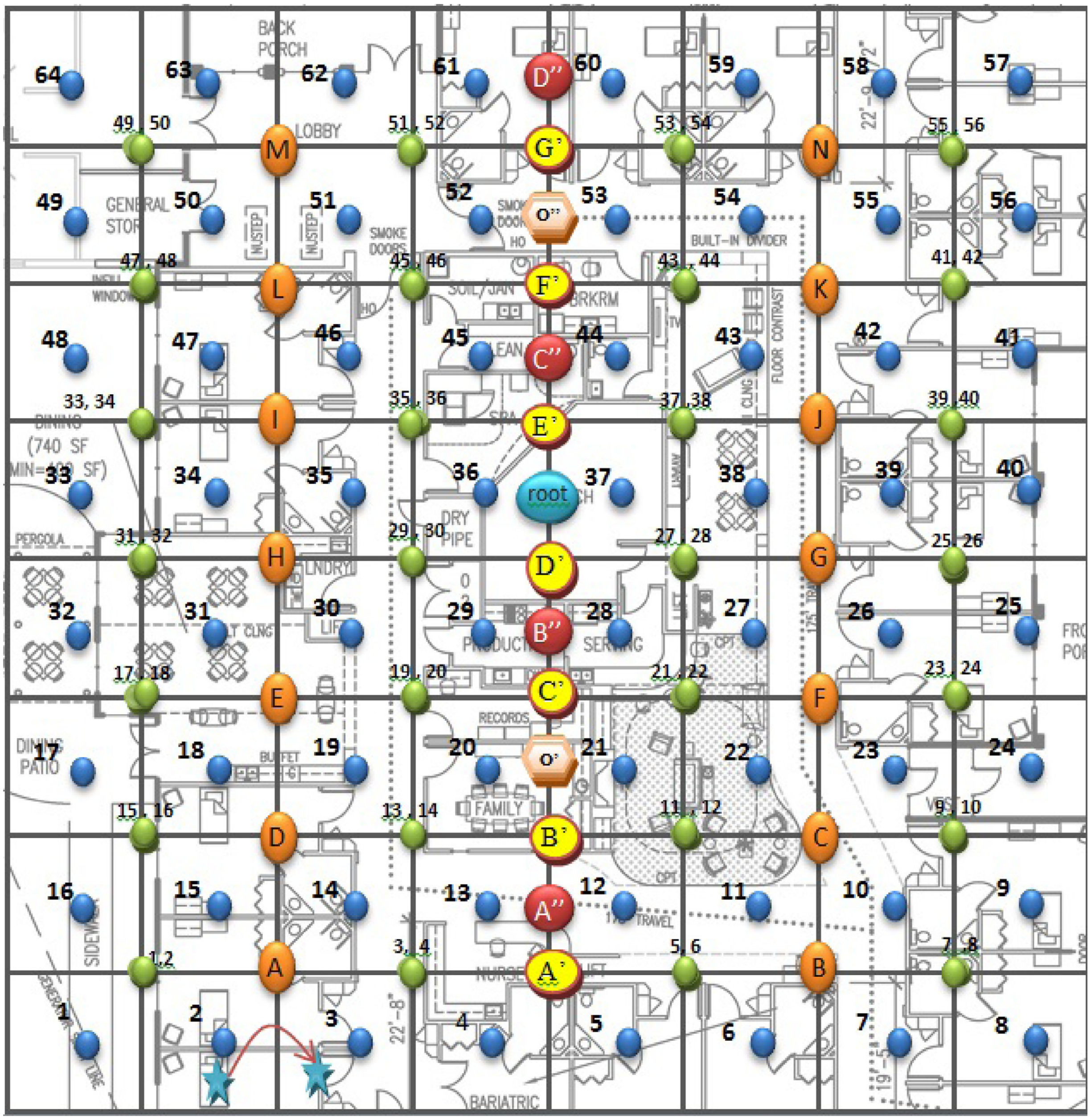

Fig. 4. Initial setup:tree construction.

Similarly, node A' is the parent of two static nodes A and B, located in the middle of the line connecting the children and so on. Each leaf node in DMP-Tree has a unique table that holds the list of the successor mobile sensor nodes. This table is used for routing and forwarding packets to mobile sensor nodes While the leaves of DMP-Tree are responsible for handling movement of mobile sensor nodes and receiving/sending data of mobile sensor node(s) to/from their parents, the non-leaf nodes act as intermediate devices to receive and send data between their parents and children. For example, in Fig. 4 when a mobile sensor node is in cell 1, the static node 1 forwards the data d of the mobile sensor node to node (1, 2) which sends it to node A. Subsequently, it will be forwarded to node A', then it is sent to node A'' which forwards it to node o' and finally it is received at the root. For simplicity, we use literal as the ID of the tree's nodes in Fig. 4.

As shown in Fig. 4, we assume that the monitoring area is divided into n equal sized cells. In Eq. (2), n is determined according to the size of the monitoring area, M, and the transmission range.

$$n = \frac{M}{L \times L} \longrightarrow n = \frac{2 \times M}{R \times R} \tag{2}$$

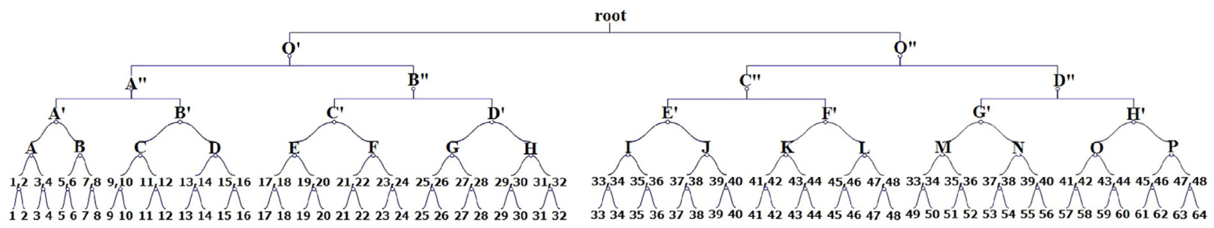

Fig. 5. DMP-Tree for the monitoring area of Fig. 4.

3.1.3. Collecting movement data

For each static sensor node at the center of each cell, the current candidate node, and the next candidate node after patient's movement, their corresponding rooms, and a time stamp is sent to the gateway at training phase. Each static sensor sends data to the predecessors in DMP-Tree until it reaches the gateway.

3.2. Movement prediction

In Section 3.2, we explain the details of the movement direction prediction approach. In Section 3.2.1, the way we construct the model is described. In Section 3.2.2, it is depicted how the generated data models are distributed after the training phase over the leaves of the DMP-Tree (Fig. 5).

3.2.1. Constructing movement prediction model

The movement prediction is achieved by modeling the data that is collected from the movement of a given patient during the training phase. Considering a sequence of states and a sequence of observations (output) at a time, HMM can predict the next state and observation. A typical first order hidden Markov model has four parameters: the hidden states (q), the initial probability (Π) which is a matrix consisting the likelihood of being in each state at initial time, state transition probability matrix (A) which are the matrix consist of the probability of transition from one hidden state to another one, and the emission probability matrix (B) which are the matrix consist of likelihood of observing particular output b in certain state a.

In our monitoring area (nursing house), each room is considered as a state. We assume that each room consists of one or more cells. The non-room parts such as a corridor are part of one of the neighboring rooms. In DMP-IOT, the observations are the cells of the monitoring area in which the moving object is located and moves. The sequence of observations corresponding to the patient's movements in a building should be collected at the training phase to build the model. We assume that the patient's current state is not only related to the previous state at time $t-1$, but it also depends on the state at time $t-2$. Thus, we have applied second-order HMM technique for modeling movement patterns. According to this model, the probability of being in a given place depends on previous two states at time $t-1$ and $t-2$.

During the training phase, the likelihood of moving from each cell to the neighboring cells (go to other observations) is computed. At training period, in each cell, the next observation with the most probability could be determined in o(1). The initial probability of transition is determined based on the lifestyle of the people in an elderly house.

Fig. 6 shows a part of the HMM model that is used by static node *2* to predict the movement direction of the patient when he is located in cell 2. In Fig. 6, q_n is the nth hidden state (room) and O_{q_n} is the output or cell that the patient stays in when the patient's state (room) equals to q_n. Considering the hidden state at time $t-2$ and previously hidden state at time $t-1$, the hidden state at the current time can be determined. Then, it enables determining the most probable observation. For example, in Fig. 6, it is assumed that the patient has been stayed in states (rooms) R_1 and R_2, at time $t-2$ and $t-1$, respectively. If the predicted hidden state at time t is R_3, the patient will move to the cell that $Sensor_3$ is located in its center.

To find unknown parameters of HMM model, the Baum-Welch algorithm is applied iteratively until the parameters in two successive iterations change less than a predefined threshold and convergence occurs. The Baum-Welch algorithm takes the initial estimates of the parameters and recalculates them using forward and backward routines (with the aid of forward and backward probability) [26]. The aim is to obtain an accurate model with precise emission probability which can best predict the future movements of a moving object. The initial probability of hidden state q_i, π_i, is estimated by Eq. (3):

$$\pi_i = pr(q_0 = room_i), \tag{3}$$

where q_t is the initial probability of being in each room (state) at time t. a_{ij} in Eq. (4) is the probability of going from $room_i$ to $room_j$. Each room consists of one or more cells.

$$a_{ij} = pr(q_t = room_j | q_{t-1} = room_i) \tag{4}$$

The second-order transition probability is computed by Eq. (5). Eq. (5) determines the probability that a patient is in $room_k$ at time t when he was in $room_i$ at time $t-2$ and at time $t-1$ the patient was in $room_j$.

$$a_{ijk} = pr(q_t = room_k | q_{t-1} = room_j, q_{t-2} = room_i) \tag{5}$$

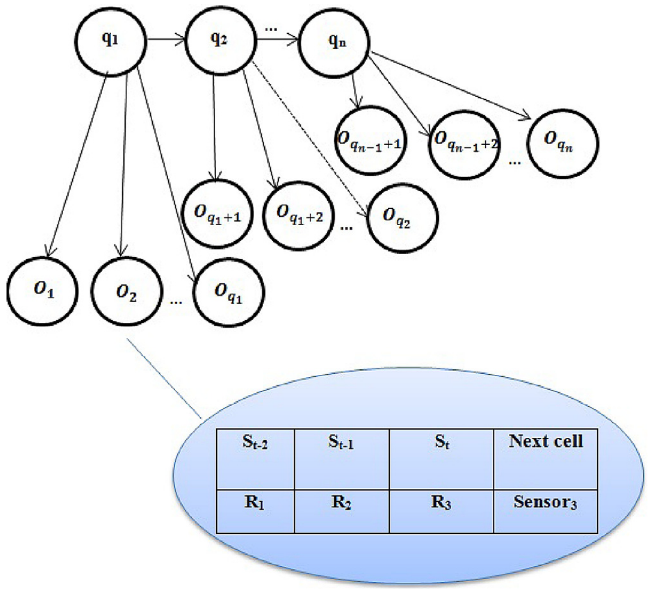

Fig. 6. An abstract view of the proposed HMM model.

The probability of being in each cell is determined by Eq. (6).

$$b_{ij} = pr(Node_t | q_t = room_j, q_{t-1} = room_i) \tag{6}$$

where b_{ij} is the probability of being in $Cell(Node_t)$ when the patient is in $room_j$ and $room_i$ at time t and $t-1$, respectively. $Cell(Node_t)$ specifies a cell in which the patient is located at time t and $Node_t$ is the static sensor node at the center of the cell that the patient is staying. As shown in Eq. (7), $pr(Node_{t-1})$ is the probability in which the patient was located in one of the neighboring cells at time $t-1$, where P_t is the probability of going from $Cell(Node_{t-1})$ at time $t-1$ to $Cell(Node_t)$ at a time t. In other words, Eq. (7) shows the probability that a patient is in a cell that $Node_t$ is located in its center at time t regarding that the patient was at $cell(Node_{t-1})$ at time $t-1$.

$$pr(Node_t) = pr(Node_{t-1}) \times P_t \tag{7}$$

With our proposed scheme, the two neighboring cells ($Cell(Node_t)$) and $Cell(Node_{t-1})$) in DMP-IOT scheme have at least one common node that is in their range. As an example, consider Fig. 4, where the common node of two neighboring nodes *1* and *2* is node (*1,2*) and the common node of two neighboring nodes *2* and *3* is node *A*. The common node of two given nodes is determined by Eq. (8):

$$\begin{aligned} pr(Cell(Node_t)) \cap pr(Cell(Node_{t-1})) = node_{common} :\in Range(cell_{Node_t}) \cap cell_{Node_{t-1}} : \\ node_{common} \in ancestor(Node_t, Node_{t-1}) \vee \\ node_{common} = (parent(Node_t) \vee parent(Node_{t-1})) \end{aligned} \tag{8}$$

The complete second-order HMM model is shown in Eq. (9):

$$\lambda = (\pi_i, a_{ij}, a_{ijk}, b_{ij}) \tag{9}$$

Then, the forward and backward probability must be calculated. The forward probability α is the probability that a patient was in room $room_1, room_2, \ldots, room_{i-1}$ while being in $room_i$ at a time *t*. $room_i$ is the *i*th room. Using forward routine, α is estimated by Eqs. (10), (11), and (12), respectively. *T* is the total number of the movements in the training phase.

$$\alpha_1(i) = \pi_i b_i(Node_1) \tag{10}$$

$$\alpha_t(j,k) = \sum_i \alpha_{t-1}(i,j) a_{ijk} b_k(Node_t) \quad , 3 \le t \le T \tag{11}$$

$$\alpha_t(k) = \sum_j \alpha_t(j,k) \;\; 3 \le t \le T \tag{12}$$

The backward probability parameter $b_k(Node_t)$ in Eq. (11) is the probability of being in a cell centered by $Node_t$ and being in $room_k$ at time t which is computed by Eq. (13). The common node, $node_{common}$, is determined by Eq. (8). *Rooms* is the set of all rooms in the monitoring area.

$$b_k(Node_t) = \sum_s b_s(Node_{t-1}) \times P_t, \;\; s \in Rooms, \;\; b_k(Node_t) \cap b_s(Node_{t-1}) = node_{common} \tag{13}$$

Table 2
The training algorithm.

Model Training Algorithm
1.C=0 (c is the number of current iteration)
2.Initialize $\lambda = (A, B, \pi)$
3.Compute $\alpha_1(i), \beta_1(i), \alpha_t(i), \beta_t(i)$
4.Adjust the model: compute $\overline{A}, \overline{B}, \overline{\pi}$
5.C=c+1
6.Check the convergence condition; if it is not met, go to 3

The backward probability β is approximated by using second-order transition using Eqs. (14) and (15).

$$\beta_T(i) = \begin{Bmatrix} 1 & \text{for the final room that the patient stays} \\ 0 & \text{otherwise} \end{Bmatrix} \tag{14}$$

$$\beta_t(j) = \sum_i a_{ji} b_i(Node_{t+1}) \beta_{t+1}(i) \tag{15}$$

In Eq. (15), $b_i(Node_{t+1}) = \sum_s b_s(Node_t) \times P_t$. As shown in Eq. (8), it is the probability of observing $node_{common}$ on the common line of two cells where $node_{common}$ could be the ancestor of both nodes or the parent of only one of them. For instance, the common node of two nodes *24* and *25* is the node (*23,24*) which is the parent of node *24*. The common node of two nodes *6* and *7* is node *B* which is the ancestor of the both nodes.

In the next step, the parameter of the model is recalculated. Eq. (16) shows the re-estimation of the initial state probability.

$$\overline{\pi_l} = \frac{\alpha_1(i)\beta_1(i)}{\sum_i \alpha_1(i)\beta_1(i)} \tag{16}$$

In Eq. (16), $\alpha_1(i)$ is the initial probability of being in each cell when the patient is in room $room_i$ and $\beta_1(i)$ is the probability of being in a sequence of neighboring rooms from time 2 until T when the patient is in room i at the initial time. T is the total training time.

$$\overline{a_{ij}} = \frac{\alpha_1(i) a_{ij} b_j(Node_2) \beta_2(j)}{\sum_j \alpha_1(i) a_{ij} b_j(Node_2) \beta_2(j)} \tag{17}$$

$\overline{a_{ij}}$ in Eq. (17) represents the probability of transition from $room_i$ to $room_j$ divided by the expected number of times a patient goes from $room_i$ to any neighboring rooms. Eq. (18) shows the probability of transition from room q_j to q_k if the room that the patient is there before $room_j$ is $room_i$.

$$\overline{a_{ijk}} = \frac{\sum_{t=1}^{T-2} \alpha_{t+1}(i,j) a_{ijk} b_k(Node_{t+2}) \beta_{t+2}(k)}{\sum_{t=1}^{T-2} \sum_k \alpha_{t+1}(i,j) a_{ijk} b_k(Node_{t+2}) \beta_{t+2}(k)} \tag{18}$$

$$\overline{b_j(k)} = \frac{\sum_{t=1, Node_t = v_k}^{T} \alpha_t(i)\beta_t(i)}{\sum_{t=1}^{T} \alpha_t(i)\beta_t(i)} \tag{19}$$

$\overline{b_j(k)}$ in Eq. (19), is the probability of being in a cell centered by node $Node_k$ in $room_i$ divided by the expected number of times in which the patient is in $room_i$. $node_{common}$ is determined in Eq. (8). As mentioned earlier, we choose second-order HMM assuming that the probability of each movement depends on the room where the patient stays as well as two rooms he stayed before at time $t-1$ and $t-2$. This happens because the previous location of the patient affects the next place she/he might walk. The algorithm of model training at the gateway is shown in Table 2. *A*, *B* in Table 2, as defined in the first paragraph of this section are the matrices of transition and emission probability, respectively. During the training and data collection phase, the movement direction detection in the network is done by calculating RSSI along with LQI.

The algorithm for testing the model is shown in Table 3. According to this algorithm, the most probable hidden state is determined in each step. The most probable output(cell) is estimated, accordingly.

Table 3
The algorithm for Applying and testing the constructed model.

```
Testing Model Algorithm
0. t=1
1. Choose hidden state $q_i$ at time t maximizing $\alpha_T(i)$
2. Choose next hidden state $q_j$ maximizing $\overline{a_{ij}}$
3. Choose next hidden state $q_k$ maximizing $\overline{a_{ijk}}$
4. Predict next observation $Node_k$ maximizing $\overline{b_j(k)}$
5. If the process continuous, then i=j, j=k , and go to 3
```

Table 4
Movement management algorithm (movement detection, identifying movement direction, recovery from false prediction).

```
Recovery Algorithm
1. Candidate node detects the movement of mobile sensor node
2. Candidate node sends $NN$ message to the predicted candidate node
3. Predicted candidate node sends $CandidateReq$ message to the mobile sensor node
4. If mobile sensor node Receives $CandidateReq$ message then
5.     mobile sensor node sends $CandidateRes$ message to the predicted candidate node $i$
6.     $i$ sends $NNACK$ to the previous candidate node
7.Else
8.     It broadcasts $FindCandid$ message containing $(PevCandidatenodeID, PredictedCandidatenodeID)$
9.     Any static node $i$ that receives the $FindCandidReq$ message from mobile sensor node calculate RSSI
10.    If the calculated RSSI is above some threshold
11.        they send $CandidateResponse$ message to the mobile sensor node
12.        i:the static node with the highest RSSI sends $NNACK$ to previous candidate node
13.        PrevCandidate node sends the previous and current hidden state (room) to $i$
14.    EndIf
15. EndIf
```

3.2.2. Distributing data model

After the HMM model is constructed in the training phase, it must be divided into sub-models and each sub-model should be downloaded into the corresponding static leaf node. The sub-model of each leaf node holds data required for deciding the next probable cell to move at each time slot. The leaf static node in each cell holds the sub table, so-called Hidden State table (*HSTable*) with the following columns: previous hidden state (PHiddenState), current hidden state (CHiddenState), next hidden state (NHiddenState), and next observation (nextObservation).

In Fig. 4, the initial probability is generated based on the patient's daily tracking in the elderly house. The table has $8 \times 8 \times 8$ memory size at maximum.

The maximum number of neighbors for each cell is eight. This happens because each room has eight rooms in neighbors. Thus, in second-order HMM that each state is dependent on two previous states, it is multiplied by 8, two times. As we mentioned earlier, we assume the number of hidden states equals to the number of the rooms in the monitoring area.

A *HS* sub-table is forwarded to each leaf node. For each leaf node *a*, the sub-table contains those rows of the *HSTable* for which their *nextObservations* column value is *a* (assume *NHiddenState* is *sn*) in addition to the rows whose *CHiddenState* value is equal to *sn*.

3.3. Movement management

In this section, we only focus on part of the mobility management that is related to movement prediction. Table 4 presents the movement management algorithm (movement detection, identifying the movement direction and recovery from false prediction parts). The following sub-sections correspond to the parts of the presented algorithm.

3.3.1. Movement detection

In order to detect the movement of a mobile sensor node, the candidate node sends a message to the mobile sensor node in a specific time interval and computes RSSI. Whenever the RSSI of the message comes below a threshold, the candidate node detects that the mobile sensor node is about to move (Line 1) of Table 4.

3.3.2. Identifying movement direction

When the current candidate node detects that the mobile sensor node is about to move, it sends the current hidden state value and the next hidden state (room) to the next probable cell (new candidate node). These two values are required to predict the direction of the patient's next movement. According to these two values, the current candidate node determines

Table 5
The parameters of the experiment.

Mobility model	Mobile sensor node speed (m/s)	Mobile sensor node count	Mobile sensor node direction
Random	0.5–10	128	0–2π

the new candidate node by looking up in *HSTable* for determining the next hidden state (room) and the ID of the new candidate node of the mobile sensor node in the next movement. The previous candidate node sends *NN* message to the predicted candidate node (Line 2) to wake it up. The current candidate node also sends the current and next hidden state value to the predicted candidate node. When the predicted candidate node receives *NN* message, it sends a *CandidateReq* message to the mobile sensor node (Line 3). If the mobile sensor node receives the *CandidateReq* message it sends *CandidateRes* to the predicted candidate node and it sends *NNACK* message to the previous candidate node (Line 5–6).

3.3.3. *Recovery*

To compensate false prediction, DMP-IOT contains an innovative recovery mechanism, depicted in Table 4. When the mobile sensor node does not receive a message from the predicted candidate node for specific time threshold, it detects a false prediction and the recovery process is started (line 9–12). If the mobile sensor node does not receive *CandidateReq* message during a specific time, it broadcasts *FindCandidReq* message containing the ID of previous candidate node (Line 7–8). The neighboring static nodes that receive this message, considering that the RSSI of the message is above a certain threshold, send *FindCandidRes* to the mobile sensor node. At the mobile sensor node side, a node with greatest RSSI is chosen (based on the received messages) to be a new candidate node of the mobile sensor node and sends *NNACK* message to the previous candidate node (Line 9–12). The previous candidate node sends the previous and current hidden state (room) to the new identified candidate node as this data is required for predicting the next cell in the next movement of the mobile sensor node (Line 13).

4. Simulation

To evaluate DMP-IOT, we have simulated the environment using Cooja [27]. There are various COTS operating systems implemented for low-power wireless networks. Among them, TinyOS and Contiki are the most popular as they provide various functionalities [28]. The Contiki operating system is initially designed for IP-based networks and it is well-known as Internet of Things emulator [29]. It has more facilities and extensions for IP-based protocols. Cooja is Java-based simulator developed for simulations of sensor nodes running Contiki operating system. The simulation parameters are shown in Table 5.

The movement data of a given patient is synthetically generated after studying the lifestyle of several elders. Whenever a patient moves from one cell to another, the data is sent to the gateway. Thus, the time resolution is the minimal duration of staying in one cell. For each static sensor node at the center of each cell, the current candidate node, the next candidate node after patient’s movement, their corresponding rooms, and a time stamp is sent to the gateway at training phase. Each static sensor sends data to the predecessors in DMP-Tree until it reaches the gateway.

At the first step, we compare the accuracy of second-order HMM movement prediction with the ANN. The input of the ANN is the last cell where the patient is entered, the number of considered hidden layers is 7,8, and 9, and the output of the neural network is the predicted cell where the patient will choose in the next movement. The learning rate is 0.2, and the number of backward steps is 10. The best prediction accuracy is achieved when the number of hidden layers is set to be 8. Fig. 7 shows the total prediction accuracy regarding different cell sizes for both HMM and ANN. The cell size demonstrates the granularity of observations. As it is depicted in Fig. 7, the best prediction accuracy that we achieved is when the cell size is about 50 m^2. With this cell size, the average prediction accuracy that we achieved is 83%. With the same cell size, the best achieved prediction accuracy is 94%. The best result is obtained when the lifestyle of the moving object in the training phase is very similar to the one during the prediction period. As shown in Fig. 7, if the cell size is too small or too large, the false prediction rate increases. If the cell size becomes too small, it means that in each room there could be at least two cells. Since in each state (room), the probability of a single observation (cell) is more than the other cells, a larger number of cells in a room results in a higher false prediction rate. If the cell size becomes larger, predicting the next movement becomes more complicated as the next movement covers a larger area and predicting the exact location in larger area results in increasing the false prediction rate. Furthermore, larger cells may not be covered by a single static node at the center of the cell, and it also increases the false prediction too.

We have compared the handoff cost of DMP-IOT with MLOWPAN [9]. The reason to choose MLOWPAN is that it has a tree-like scheme like our solution and like us, the routing is done in the link layer. The movement prediction in MLOWPAN is based on AOA, assuming that each node knows its position as well as other neighbors’ positions.

Fig. 8 shows handoff delay in terms of prediction accuracy. Our proposed method has less delay compared to MLOWPAN. This happens because if the patient suddenly changes his movement direction, The mobility management approaches that use AOA method (such as MLOWPAN) may fail to detect the correct direction. Furthermore, due to noise and environmental factors, the direction prediction of AOA method may not always be correct. It is worth mentioning that it needs to exchange

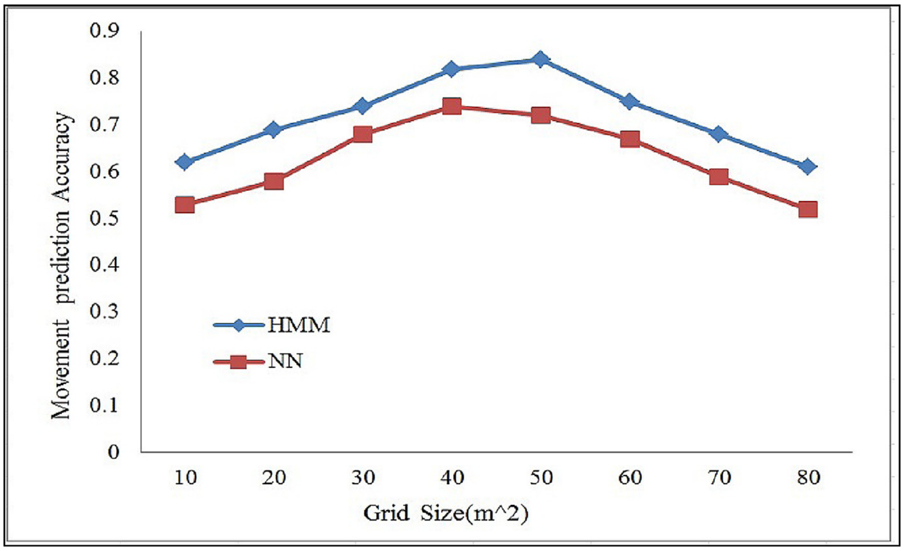

Fig. 7. Total prediction accuracy regarding cell size.

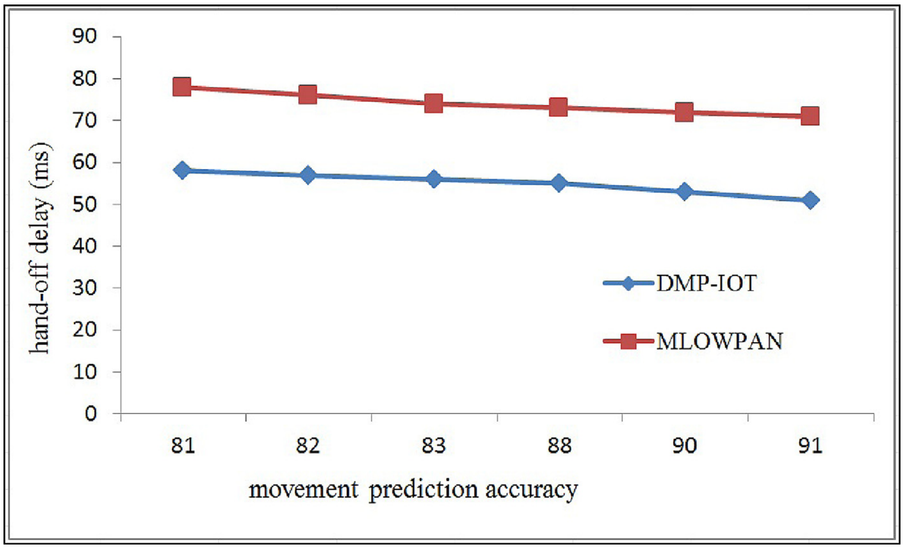

Fig. 8. Delay regarding movement prediction accuracy.

more packets for computing the correct orientation. The cost of a wrong prediction includes delay and packet loss because the right direction of the mobile sensor node is lost.

Fig. 9 shows packet loss in terms of prediction accuracy. As shown in Fig. 9, due to the delay caused by false prediction, whenever the false prediction increases, the packet loss grows, as well. The wrong prediction happens when the patient changes the direction suddenly or when the speed is high, and there is not enough time that AOA runs, accurately. Noise and environmental factors also have a great negative impact on the AOA method. But due to our recovery mechanism, this delay is limited, and it is less than MLOWPAN algorithms.

Fig. 10 shows handoff cost (delay) regarding the patient's speed. As shown in the figure, our algorithm works significantly better than MLOWPAN as the speed of patient increases. This happens because in high speeds, determining the direction of movement with AOA method cannot be accurate.

Fig. 11 shows the total handoff cost in bits regarding false prediction rate. As can be seen, if the false prediction rate grows, the handoff cost increases as well. This happens because the recovery procedure is invoked more frequently, which consequently causes more message exchanges between the mobile sensor node and the corresponding neighboring leaf static nodes. However, the handoff cost is less than MLOWPAN since AOA also depends on environmental factors including noise. For MLOWPAN, this cost is constant because it does not have a recovery mechanism whenever false prediction of the movement direction of mobile sensor nodes occurs.

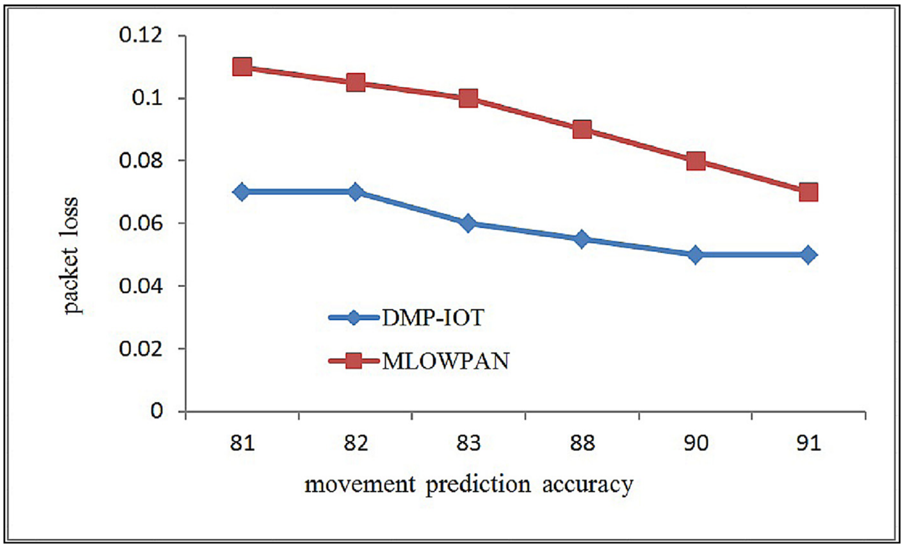

Fig. 9. Packet loss regarding movement prediction accuracy.

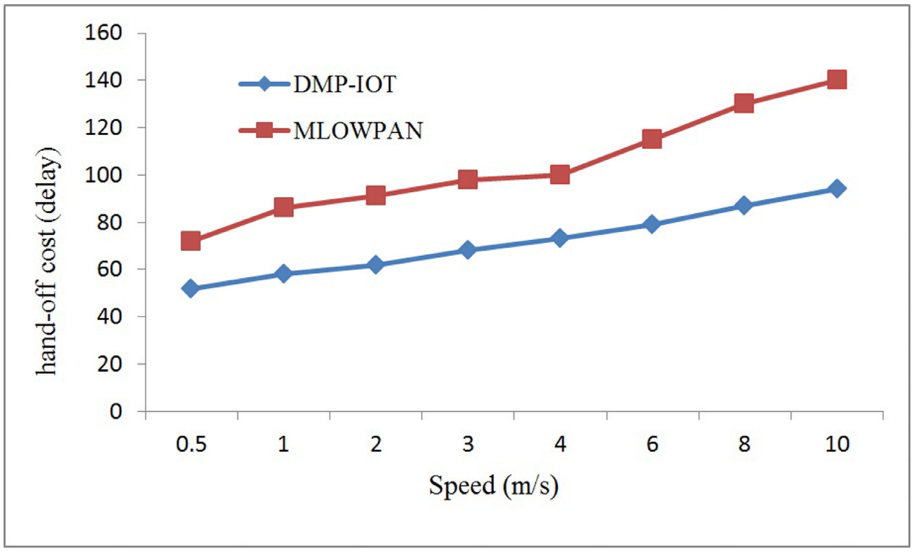

Fig. 10. Handoff cost (delay) regarding speed.

Fig. 12 shows the average prediction accuracy regarding different time resolutions for each of HMM and ANN methods. We can sample the patient's position its cell number, in different time resolutions. The minimum value between the sampling time and the patient's presence duration in a cell is defined as time resolution. In this paper, we have sampled the patient positions whenever he goes to another cell. But the resolution can be larger or smaller. As it is shown in Fig. 12, if the time resolution becomes higher, the accuracy would be higher, too. The accuracy of HMM is higher than ANN. This happens because unlike ANN, the sequence of movements is considered in HMM.

4.1. Analytical verification

As mentioned in the previous section, we trained the model with the data of the patient's movement in 31 days and we have gained 83% accuracy, in average (μ). In this section, our aim is to show that the accuracy of the movement prediction would not go below 83% if we increase the population size. Here, the population is the number of days that the accuracy is measured base on. In other words, we want to show that our result is valid for the population by null hypothesis testing technique. To this end, we determine two speculations (H_0) and (H_1) and the data is analyzed with the goal of determining whether the stated speculation is unreasonable. We define H_0 and H_1 as Eq. (20):

$$H_0 : \mu \leq 83$$

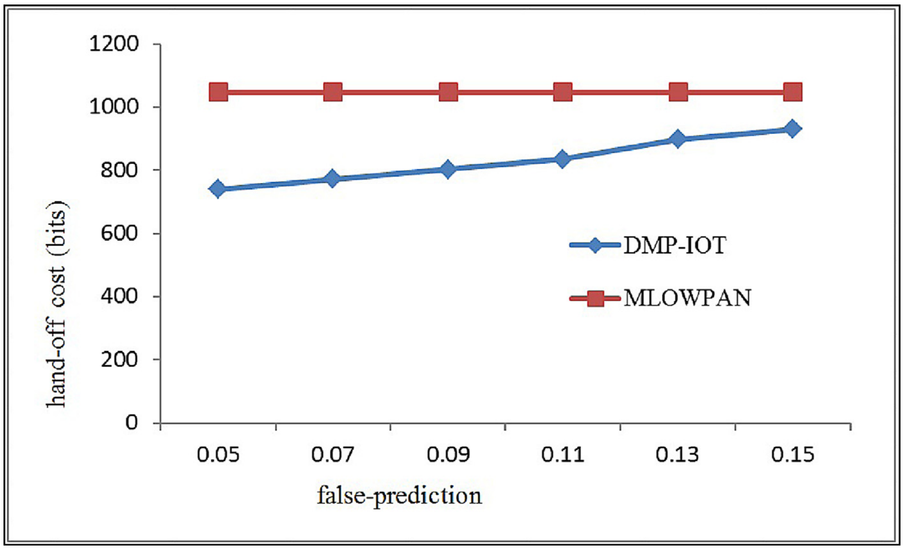

Fig. 11. Handoff cost regarding false movement prediction.

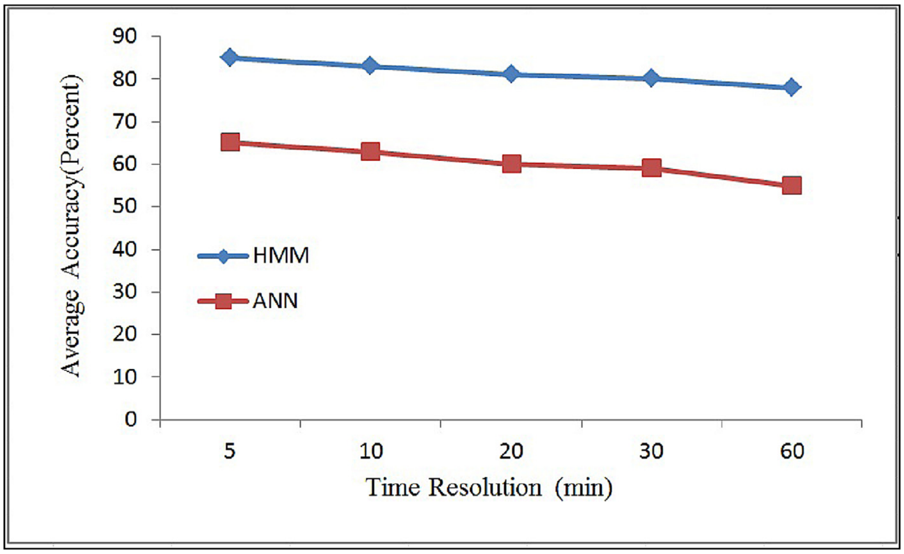

Fig. 12. Prediction accuracy in terms of time resolution.

$$H_1 : \mu > 83 \tag{20}$$

The question is: is there an empirical evidence indicating that the speculation (H_0) is probably incorrect? If we prove that (H_0) is incorrect, then the alternative speculation (H_1) which is our correct speculation, could be verified [30]. As we do not know the variance of the whole population, the distribution of our population obeys $t - Student$. We run the prediction scheme for thirty patients, hence $n = 30$. We assume Type I error α to be 0.01%. According to *Student*'s t-distribution, t distribution with degree of freedom 29 and error 0.01 is 2.462. The $t - Student$ distribution is shown in Eq. (21):

$$t_{n-1} = \frac{\overline{X} - \mu}{\frac{S}{\sqrt{n}}} \tag{21}$$

The average accuracy and variance that we achieved for movement prediction of 30 patients is 85.6 and 4.69, respectively. If we substitute the parameters in Eq. (21), it would be 3.1 which is greater than 2.462. Therefore, H_0 assumption is not correct and the prediction accuracy is more than 83%. The $t - Student$ distribution plot of our testing prediction accuracy is shown in Fig. 13.

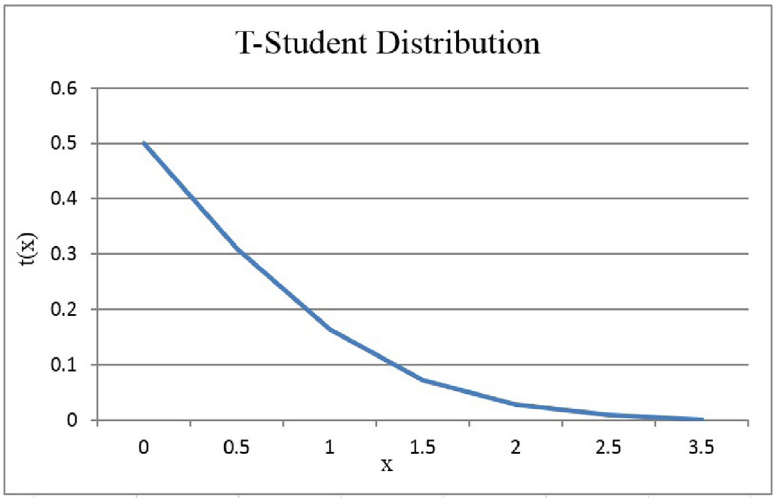

Fig. 13. The T-distribution of prediction accuracy.

5. Conclusion

In this paper, we have proposed a novel hierarchical network scheme for improving the cost and accuracy of detecting the direction of mobile sensor nodes' movement as part of the mobility management in IOT safety critical applications like health-care. The aim is to achieve a data distribution scheme which predicts the movement direction of mobile sensor nodes while simplifying routing and forwarding data of the mobile sensor node to/from the gateway. The prediction scheme is based on second-order HMM that is trained with patient's tracking data inside a building. Our proposed scheme reduces the handoff delay compared to earlier works where RSSI is measured for detecting the direction of movement. DMP-IOT removes the need for specific hardware (directional antennas, antenna arrays, etc.) which are alternative solutions for determining the movement direction (AOA). Unlike directional antenna, DMP-Tree is capable of tolerating the failure of static nodes. The proposed algorithm decreases handoff cost compared to the case that the movement direction is done based on RSSI or AOA algorithms. We gain 83% prediction accuracy on average, and we have also decreased the handoff cost 25% at a minimum, compared to the case that no prediction is made. To offset the false prediction, the suggested recovery mechanism prevents the mobile sensor node disconnection from the network.

In traditional WSNs, the current candidate node should communicate with the nodes in the neighboring cells to determine the direction of the movement imposing considerable overhead and power consumption of nodes. With the proposed movement prediction scheme, there would be no need for communication between candidate nodes. DMP-IOT eliminates the communication cost between the current and new candidate node. Furthermore, with accurate movement prediction of the mobile sensor node, the power consumption of mobile sensor node(s) as well as handoff delay significantly decreases. Since each leaf node maintains only a small part of the model, the mobility prediction is fast and profitable.

Future works include studying other factors that may have a positive influence on increasing the accuracy of the movement prediction in health-care environments. The multi-user problem and the scalability of the approach could also be considered, in the future.

References

[1] Atzori L, Iera A, Morabito G. The internet of things: a survey. Comput Netw 2010;54(15):2787–805.
[2] Khalil N, Abid M, Benhaddou D, Gerndt M. Wireless sensors networks for internet of things. In: Intelligent sensors, sensor networks and information processing (ISSNIP), 2014 IEEE ninth international conference on; 2014. p. 1–6.
[3] Meharouech A, Elias J, Mehaoua A. Future body-to-body networks for ubiquitous healthcare: a survey, taxonomy and challenges. In: Future information and communication technologies for ubiquitous healthCare (Ubi-HealthTech), 2015 2nd international symposium on; 2015. p. 1–6.
[4] Bouaziz M, Rachedi A. A survey on mobility management protocols in wireless sensor networks based on 6lowpan technology. Comput Commun 2014;52.
[5] Patwari N, Ash J, Kyperountas S, Hero A, Moses R, Correal N. Locating the nodes: cooperative localization in wireless sensor networks. Signal Process Mag IEEE 2005;22(4):54–69.
[6] Wang X, Sun Q, Yang Y. A cross-layer mobility support protocol for wireless sensor networks. Comput Electr Eng 2015;48:330–42.
[7] Wang X, Le D, Cheng H, Xie C. All-ip wireless sensor networks for real-time patient monitoring. J Biomed Inform 2014;52:406–17. Special Section: Methods in Clinical Research Informatics.
[8] Silva R, Silva JS, Boavida F. Mobility in wireless sensor networks, survey and proposal. Comput Commun 2014;52(1):1–20.
[9] Wang X, Zhong S, Zhou R. A mobility support scheme for 6lowpan. Comput Commun 2012;35(3):392–404.
[10] Zinonos Z, Vassiliou V. Inter-mobility support in controlled 6lowpan networks. In: GLOBECOM workshops (GC Wkshps), 2010 IEEE; 2010. p. 1718–23.
[11] Islam MM, Huh E-N. Sensor proxy mobile ipv6 (spmipv6)-a novel scheme for mobility supported ip-wsns. Sensors (Basel, Switzerland) 2011;11(2):1865–87.

[12] Bag G, Raza MT, Kim K-H, Yoo S-W. Lowmob: Intra-pan mobility support schemes for 6lowpan. Sensors (Basel, Switzerland) 2009;9(7):5844–77. doi:10.3390/s90705844.
[13] Ha M, Kim D, Kim SH, Hong S. Inter-mario: a fast and seamless mobility protocol to support inter-pan handover in 6lowpan. In: Global telecommunications conference (GLOBECOM 2010), 2010 IEEE; 2010. p. 1–6. doi:10.1109/GLOCOM.2010.5683658.
[14] Mathew W, Raposo R, Martins B. Predicting future locations with hidden Markov models. In: Proceedings of the 2012 ACM conference on ubiquitous computing. UbiComp '12. New York, NY, USA: ACM; 2012. p. 911–18. ISBN 978-1-4503-1224-0.
[15] Lee J-R, Han S-H, Choi Y-H. Vehicle mobility pattern-based handover scheme using discrete-time Markov chain. Comput Electr Eng 2014;40(1):100–8.
[16] Misra S, Singh S, Khatua M, Obaidat MS. Extracting mobility pattern from target trajectory in wireless sensor networks. Int J Commun Syst 2015;28(2):213–30.
[17] Zukerman I, Albrecht D, Nicholson A. Predicting users requests on the www. In: UM99 user modeling, 407. Springer; 1999. p. 275–84. ISBN 978-3-211-83151-9.
[18] Akoush S, Sameh A. Mobile user movement prediction using Bayesian learning for neural networks. In: Proceedings of the 2007 international conference on wireless communications and mobile computing. IWCMC '07. New York, NY, USA: ACM; 2007. p. 191–6. ISBN 978-1-59593-695-0.
[19] Gellert A, Vintan L. Person movement prediction using hidden Markov models. Stud Inform Control 2006;15:17–30.
[20] Baratchi M, Meratnia N, Havinga PJM, Skidmore AK, Toxopeus BAKG. A hierarchical hidden semi-Markov model for modeling mobility data. In: Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing. UbiComp '14. New York, NY, USA: ACM; 2014. p. 401–12. ISBN 978-1-4503-2968-2.
[21] Catasta M, Meratnia N, Havinga PJM, Skidmore AK, Toxopeus BAKG. Next place prediction using mobile data. In: Proceedings of the mobile data challenge workshop; 2012.
[22] Furey E, Curran K, McKevitt P. Habits: a Bayesian filter approach to indoor tracking and location. Int J Bio-Inspired Comput 2012;4(2):79–88. doi:10.1504/IJBIC.2012.047178.
[23] Duong T, Tran DQ, Tran CH. Spatiotemporal data mining for mobility prediction in wireless network. In: Proceedings of the national conference of fundamental and applied IT research (FAIR); 2011. p. 224–35.
[24] Cadger F, Curran K, Santos J, Moffet S. Location and mobility-aware routing for improving multimedia streaming performance in manets. Wireless Pers Commun 2016;86(3):1653–72.
[25] Si H, Wang Y, Yuan J, Shan X. Mobility prediction in cellular network using hidden Markov model. In: Consumer communications and networking conference (CCNC), 2010 7th IEEE; 2010. p. 1–5.
[26] Baum LE, Petrie T, Soules G, Weiss N. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. AnnMathStat 1970;41(1):164–71.
[27] Eriksson J, Österlind F, Finne N, Tsiftes N, Dunkels A, Voigt T, et al. Cooja/mspsim: interoperability testing for wireless sensor networks. In: Proceedings of the 2nd international conference on simulation tools and techniques. Simutools '09. Belgium, Belgium: ICST, Brussels; 2009. p. 1–7. ISBN 978-963-9799-45-5.
[28] Dunkels A, Gronvall B, Voigt T. Contiki - a lightweight and flexible operating system for tiny networked sensors. In: Local computer networks, 2004. 29th annual IEEE international conference on; 2004. p. 455–62.
[29] Osterlind F, Dunkels A, Eriksson J, Finne N, Voigt T. Cross-level sensor network simulation with cooja. In: Local computer networks, proceedings 2006 31st IEEE conference on; 2006. p. 641–8.
[30] Wilcox RR. Basic statistics: understanding conventional methods and modern insights. Oxford Univ Pr; 2009.

Azadeh Zamanifar is currently a Ph.D. student of Software Engineering in the Computer Engineering Department of Shahid Beheshti University. She received her M.Sc. in Computer Engineering (Software) in 2008 from Iran University of Science and Technology (IUST), Tehran, Iran. She received B.Sc. in computer Engineering (hardware) in 2002 from Tehran University. Currently, she is a Research Assistant at Self-* Laboratory, Shahid Beheshti University. Her main interest is Wireless Sensor Networks, Internet of Things and Autonomic Computing.

Eslam Nazemi received the B.Sc. degree in Computer and Research in Operation from former Planning and Computer Application School in 1977 and two M.Sc. degrees in Systems Engineering (Minor: Research in Operation) and Systems Engineering (Minor: Economy) from Isfahan University of Technology and Research and Development Institute in 1989 and 1997. He received the Ph.D. degree in Industrial Engineering (Minor: Information Technology) from the Islamic Azad University (Science and Research Branch). Currently, he is an associate professor at Shahid Beheshti University. His research interests include Autonomous Computing and Self-star concepts.

Mojtaba Vahidi received the B.S. degree in Software Engineering, Tehran Polytechnic, Iran, in 2005. He received hid M.S. and Ph.D. degree in Software Engineering from the Iran University of Science and Technology in 2008 and 2014, respectively. He is currently an assistant professor in the software engineering department in faculty of Computer Science and Engineering at Shahid Beheshti University of Tehran. His research focus is software quality with an emphasis on machine learning techniques. His other interests are Internet of Things, Human Based Computation and Gamification.